Master Thesis

im Rahmen des Universitätslehrganges „Geographical Information Science & Systems“ (UNIGIS MSc) am Interfakultären Fachbereich für GeoInformatik (Z_GIS) der Paris Lodron-Universität Salzburg

zum Thema

Wo ist die Giraffe?

Sternbilder finden mit der Microsoft Hololens

vorgelegt von

Thomas Hildebrandt, BSc
Matr. Nr.: 106722, UNIGIS Jahrgang 2020

Betreuer: Privatdozent Dipl. Ing. Dr. Gerhard Navratil

Zur Erlangung des Grades
„Master of Science – MSc”

Wien, im März 2024

Eidesstattliche Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbständig verfasst, andere als die angegebenen Quellen nicht verwendet und die benutzten Quellen beziehungsweise wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Wien, im März 2024 Thomas Hildebrandt

Keine Schuld ist dringender, als die, Dank zu sagen.
(Marcus Tullius Cicero)

Für meine Familie.
Hier auf unserer Erde oder oben bei den Sternen.

Zusammenfassung

Das Ziel dieser Masterthesis war die Entwicklung einer einfachen Softwareanwendung für die Microsoft Hololens, welche die Sternbilder (engl. Constellations) darstellen kann. Für die Umsetzung wurden unterschiedliche Sternkataloge verwendet, die beim Anwendungsstart verarbeitet werden. Die Berechnung der Sternpositionen erfolgt für die aktuelle Uhrzeit, zu welcher die Anwendung genutzt wird. Die Darstellung der Sterne erfolgt ohne separate Grafiken und wird über die Spieleentwicklungssoftware Unity umgesetzt. Die Orientierung bzw. Ausrichtung wird ohne Global Navigation Satellite System (GNSS) durchgeführt, da die Anwendung möglichst ohne zusätzliche Hard- oder Softwareausstattung genutzt werden können soll. Die Hololens verfügt über eine Inerital Measurement Unit (IMU), die für diese Arbeit ausgelesen wurde, aber nicht für die Berechnung der Orientierung miteinbezogen werden konnte. Die Darstellung der Sterne in ihrer korrekten Position konnte gezeigt werden, sofern der User an seinem Standort stehen bleibt. Die Visualisierung ist rotationsstabil, ohne Anker oder andere Techniken in ihrer Position fixiert. Zusätzliche Informationen (Sternbildlinien, Gradnetz, Beschriftungen) bringen die Hololens hinsichtlich ihrer Performance für diese Art der Visualisierung an die Leistungsgrenze. Das Problem der sich mitbewegenden Objekte, wenn sich der User mit der Hololens bewegt, also seinen Standort ändert, konnte im Zuge dieser Arbeit nicht gelöst werden.

Abstract

The aim of this master's thesis was to develop a simple software application for the Microsoft Hololens that can display the constellations. Different star catalogs were used for the implementation, which are processed when the application is started. The star positions are calculated for the current time at which the application is being used. The stars are displayed without separate graphics and are implemented using the game development software Unity. The orientation or alignment is carried out without GNSS, as the application should be able to be used without additional hardware or software equipment. The Hololens has an IMU, which was read out for this work, but could not be included for the calculation of the orientation. The display of the stars in their correct position could be shown as long as the user remains at their location. The visualization is rotationally stable, fixed in its position without anchors or other techniques. Additional information (constellation lines, degree grid, labels) push the Hololens to its performance limits for this type of visualization. The problem of moving objects when the user moves with the Hololens, i.e. changes their position, could not be solved in the course of this work.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

1. Einleitung

1.1. Aufgabenstellung und Motivation

Mit dieser Arbeit soll geprüft werden, ob die Darstellung von Sternbildern mit der Hololens von Microsoft sinnvoll möglich ist. Die Visualisierung von Sternbildern mit Techniken der Augmented Reality (AR) oder Virtual Reality (VR) ist nicht neu und kann in zahlreichen Anwendungen, beispielsweise Apps für Android oder iOS mit Unterstützung der integrierten Sensorik und Kameras, erprobt werden. Der Versuch, die Sternbilder mittels der von Microsoft so genannten Mixed Reality (MR) - Technik darzustellen ist eine Ergänzung zu bestehenden Lösungen. Die Motivation für diese Arbeit begründet sich auch auf der Idee, die Astronomie durch die Verwendung neuer Techniken den Menschen zugänglicher zu machen. Durch die Darstellung von Sternen bzw. Sternbildern mittels einer AR-Brille wird eine einprägsame Möglichkeit geschaffen, wissenschaftliche Erkenntnisse auf einfache Weise zu vermitteln. Hinzu kommt, dass für Menschen im städtischen Lebensbereich durch die zunehmende Lichtverschmutzung der Blick auf die Sterne durch das städtische Licht eingeschränkt wird. Die Verwendung einer AR-Brille kann helfen, die Sterne für die Orientierung am Himmel sichtbarer zu machen.

1.2. Forschungsfragen

Die allgemein zu klärende Frage ist, ob eine Visualisierungstechnik, die für das unmittelbare Agieren mit Objekten im Nahbereich des Benutzers gemacht wurde, auch dazu geeignet ist, weit entfernte Objekte lagerichtig darzustellen. Die Sterne des Nachthimmels stellen sich für den Betrachter als kleine Lichtpunkte dar, die als Hologramm in der Hololens bei äquivalenter Größe schwer zu erkennen sind. Im Zuge der Arbeit ist daher eine bestimmte Mindestgröße für die Hologramme zu ermitteln. Weiters soll geprüft werden, ob die Bedingungen bei Tageslicht für diese Anwendung ausreichend sind oder ob eine gut sichtbare Visualisierung nur bei dunkler Umgebung ermöglicht werden kann. Wesentlicher Aspekt im Sinne der Erprobung ist auch, ob die Orientierung des Blickfelds durch den Beobachter mit ausreichender Genauigkeit bestimmt werden kann. Als letzter Aspekt ist die Thematik der Performance von Bedeutung: die Sternbilder enthalten viele einzelne Objekte, die durch Linien verbunden werden können. Die große Anzahl der Objekte und die nötigen Polygone für eine gewohnte, dem Benutzer bekannte Darstellung benötigt entsprechende Rechenleistung. Ob diese dafür ausreichend ist, soll im Zuge dieser Arbeit geprüft werden.

1.3. Struktur der Arbeit

Diese Masterthesis gliedert sich in folgende Kapitel:

Im Kapitel Einleitung wird die Aufgabenstellung dieser Masterthesis erläutert. Im Abschnitt Sternbilder, Mixed Reality und die Hololens werden allgemeine Informationen zu den Sternbildern generell und wenige historische Aspekte erklärt. Weiters werden die Unterschiede zwischen den einzelnen Visualisierungstechniken besprochen und die Hololens von Microsoft vorgestellt. Das dritte Kapitel mit dem Titel Forschungsstand gibt einen Überblick über die aktuelle Situation im Bereich der Mixed Reality - Forschung. Der nächste Abschnitt namens Theoretische Grundlagen widmet sich den Grundlagen und Herleitung der Koordinatenbestimmung. Im Abschnitt

Datenquellen und Positionsberechnung wird auf die verwendeten Datenquellen eingegangen und die Berechnung der Positionen beschrieben. Das Kapitel Anwendungsprogrammierung widmet sich der Erstellung des Demonstrators mit den Daten aus dem vorangegangenen Kapitel. Danach müssen die Ergebnisse geprüft und mit dem Anforderungskatalog, der dieser Arbeit zu Grunde liegt, verglichen werden. Die abschließende Diskussion widmet sich dem Ergebnis dieser Arbeit und beleuchtet das Verbesserungspotential, das Fazit als letztes Kapitel dient zur Bilanzierung.

2. Sternbilder, Mixed Reality und die Hololens

2.1. Die Sternbilder

Die Sonne, der Mond und die Sterne waren vor dem Zeitalter moderner Navigationssysteme wichtige Hilfsmittel zur Orientierung.

> „Sternbilder sind ein historisches Bezugssystem, mit dem sich Positionen am Himmel beschreiben lassen."

Dies ist die Einleitung aus dem Buch von Hoffmann (2021) und beschreibt den Kern der Verwendung aus historischer Sicht, wenn man sich auf Fragen der Navigation beschränkt. Betrachtet man die historische Bedeutung der Sternbilder für die Menschheit, ist ein Blick weiter zurück in die Entwicklungsgeschichte von Nöten, um die Zusammenhänge besser einordnen zu können. Die Sternbilder, wie sie in ihrer populären Form namentlich existieren, wurden 1922 auf der ersten Generalversammlung der neu gegründeten International Astronomical Union (IAU) in Rom definiert:

> „At its first General Assembly held in Rome in 1922, the IAU's Commission on Notations and Units agreed on a list of 88 constellations covering the entire sky, with three-letter abbreviations of their Latin names." (IAU, 2023)

Entgegen der landläufigen Meinung, dass diese 88 Sternbilder über ihre Linien-Form definiert sind, sind sie von der IAU über rechtwinkelige Koordinatenbereiche auf der Himmelssphäre definiert worden. Der belgische Astronom Eugène Delporte wurde in den 20er Jahren des 20. Jahrhunderts damit beauftragt, die Grenzen der Sternbilder festzulegen. Die formale Definition bzw. die Anerkennung durch die IAU erfolgte 1928. Seit damals sind die Grenzen bestimmt, die Positionen beziehen sich auf die Epoche B1875. Wir greifen hier als Beispiel, und in weiterer Folge immer wieder, auf das Sternbild Giraffe zurück. Dieses Sternbild ist über die Koordinaten in Tabelle 12 definiert und hat gemäß dieser Koordinaten die Form nach Bild 1. Die Figur am Sternenhimmel stellt sich aber dar wie in Bild 2 skizziert.

Die Entwicklung der Sternbilder geht weit zurück in die Vergangenheit. Man kann davon ausgehen, dass die ersten Muster parallel zur Beobachtung der Sterne definiert wurden. Erste Hinweise finden sich in der Höhle von Lascaux. Dort wurden in den 40er Jahren des vorigen Jahrhunderts Felsmalereien entdeckt, die Rückschlüsse auf die Beobachtung der Sterne und das Definieren von Mustern zulassen. Das Alter der Felszeichnungen wir auf ca. 15.000 Jahre geschätzt und die dort in den Fels geritzten Muster deutet man als eine Konstellation um Cygnus (Künzl, 2005). Allerdings ist dies eine Interpretation der dortigen Felsmalereien, gesicherter Nachweis ist das Bild naturgemäß nicht, denn es gibt keine überlieferten Beschreibungen oder sonstige Hinweise, die diese Theorie untermauern.

Um die Bedeutung und die überlieferten Bezeichnungen (unabhängig der heutigen Gültigkeit) besser einordnen zu können, muss man berücksichtigen, dass die Sternbilder eine große mythologische Bedeutung in der früheren Zeit hatten. Im Anhang sind mit Tabelle 19 die Namen der aktuell definierten Sternbilder gemäß IAU angeführt. Die Sternbilder der nördlichen Hemisphäre haben einen babylonischen, griechischen oder auch römischen Hintergrund, viele Sternbilder der

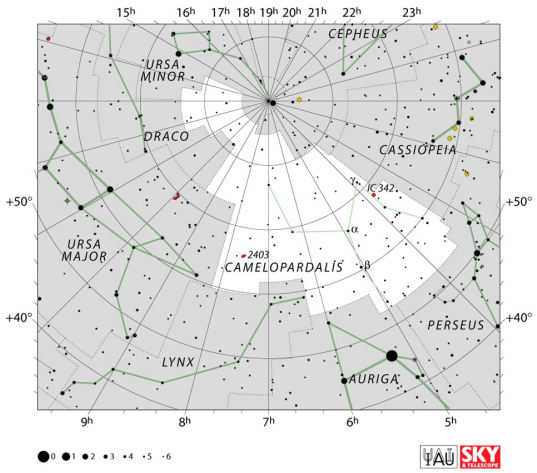

Abbildung 1: Die Grenzen des Sternbilds Giraffe (lateinisch Camelopardalis) gemäß IAU. Die weiße Fläche definiert das Sternbild Giraffe, die grüne Linie innerhalb der weißen Fläche markiert wichtige Sterne und ein gängiges Muster diese Sternbilds. Quelle: iau.org

südlichen Hemisphäre besitzen eher technische Namen (Luftpumpe), dies geht auf die jüngere Vergangenheit der Entdeckung und Bestimmung zurück.

Das Festlegen von Mustern am Himmel setzt gleichermaßen das Katalogisieren von Sternen voraus. Erste Hinweise auf Sternkataloge zeigen gebrannte Tontafeln aus der Zeit der Babylonier, datiert auf etwa 2000 Jahre vor Christus. Das klassische Beispiel aus dieser Zeit ist das Werk MUL.ALPIN. Diese zwei Tontafeln, etwa jeweils die Größe einer herkömmlichen A4-Seite, enthalten eine Liste von Auf- und Untergangszeiten von Gestirnen und eine Beschreibung des Sternbilds Pflug. Auf diesen Tafeln ist auch die Basis für ein sexagesimales Koordinatensystem (360°) ersichtlich, d.h. im alten Mesopotamien wurde bereits das heute noch gebräuchliche System kreiert. Datiert werden die Tafeln, je nach Quelle und Interpretation, auf 700 v. Chr. oder noch einige Jahrhunderte davor. Aufgrund der drauf enthaltenen Mondbahn lässt sich auch der Vorläufer des Zodiak-Kreises erkennen. Die griechische Sternbildkultur fußt auf Notizen von Eudoxos und werden auf das 4. Jahrhundert vor Christus datiert. Aus damaliger Zeit ist auch gesichert, dass die Ekliptik, also die scheinbare Bahn der Sonne um die Erde, bekannt war. Der Zodiakalkreis, eingeteilt in 12 gleiche Abschnitte (eigentlich 13), ist seit damals etabliert. Aus der griechisch-römischen Geschichte entspricht der Altas Farnese dem Stand der damaligen Zeit. Dieser wird auf das 2. Jahrhundert vor Christus datiert und zeigt Atlas (einen Gott), der einen Globus trägt. Auf diesem Globus finden sich zahlreiche Linien (Äquator, Ekliptik, u.a.) und je nach Quelle, 41 oder 42 Sternbilder aus dem griechischen Almagest. Ein wichtiges Werk aus der späteren Zeit ist der Atlas Uranometria. Dieser Atlas wurde 1603 von Johann Bayer, ein deutscher Astronom, veröffentlicht und enthält sichtbare Sterne für den Nord- und Südhimmel sowie in Summe 60 Sternbilder: jene 48 aus dem Almagest und zwölf neue Kreationen. Im Laufe der Zeit wurden neue Sternbilder bestimmt, bestehende angepasst und die

dazugehörigen Zeichnungen dem Zeitgeist angeglichen. Dies führte im Laufe der Jahrhunderte zu einem unübersichtlichen status quo und wurde erst durch die Definition der IAU in den 20er Jahren des vorigen Jahrhunderts vereinheitlicht. (Hoffmann, 2021)

Nach dieser Einführung wollen wir noch das titelgebende Sternbild der Giraffe, der guten Form halber, ausführlicher beschreiben. Die Informationen sind aus den Werken Schilling (2019) und Hoffmann (2021) entnommen. Der lateinische Name ist **Camelopardalis**, im deutschen naturgemäß Giraffe bzw. im englischen analog. Die offizielle Abkürzung lautet **Cam**. Die Fläche am Himmel gemäß der IAU Definition beträgt etwa 757 Quadratgrad, womit das Sternbild im Sinne der Größe Rang 18 einnimmt. Der hellste Stern ist β **Camelopardalis** mit einer Helligkeit von mag 4.0 (das ist relativ gering, d.h. die Sterne dieses Sternbildes sind mit bloßem Auge nur eingeschränkt sichtbar). Theoretisch befinden sich in diesem Sternbild 152 Sterne, die man von der Erde aus erkennen kann. Allerdings ist dies nur möglich an Orten ohne oder nur mit geringer Lichtverschmutzung. Das Sternbild ist zirkumpolar, d.h. in der nördlichen Hemisphäre geht es nie unter, die beste Sichtbarkeit ergibt sich im Winter. Innerhalb dieses Sternbildes befinden sich 53 NGC-Objekte, also zum Beispiel Nebel oder Galaxien. Der mythologische Hintergrund dieses Sternbildes ist weniger spektakulär als bei anderen, vielleicht populäreren Sternbildern. In seiner ursprünglichen Form war die Giraffe eigentlich als Kamel definiert und hat im Laufe der Zeit eine kleine Umwandlung erfahren. Man mag die heutige Interpretation als Giraffe als kulturelles Missverständnis interpretieren können, in letzter Konsequenz hat sich aber die Definition als Giraffe durchgesetzt. Eingeführt wurde das Sternbild 1612 vom flämischen Astronom Petrus Plancius, weil auf der Himmelssphäre schlicht noch undefinierte Flecken, die man befüllen musste, vorhanden waren. Plancius hatte dies auch auf der Südhalbkugel getan. Damals gab es neue Möglichkeiten der Erkundung der Sterne, z.B. die ersten Varianten des Teleskops, die für die Astronomen dieser Zeit vorher ungeahnte Einblicke in den Nachthimmel ermöglichten.

Hoffmann (2021) schreibt:

> ”Das lateinische Wort Camelopardalis ist eine Umschrift des griechischen Wortes für Giraffe und heißt wörtlich ”gescheckte Kamel”, was aber in römischer Zeit vermutlich bereits vergessen war. ... Plancius meinte und zeichnete jedenfalls definitiv die lateinische Giraffe und nicht das Kamel, doch das eigenartige Wort wurde bisweilen von anderen Astronomen als Kamel missverstanden.”

Im Sinne einer mythologischen Deutung gibt es durch die verhältnismäßig kurze Zeit, die das Sternbild besteht, keine historische Verbindung. Man kann die Giraffe als missverstandenes Kamel einer religiösen Deutung zuführen, aus christlicher oder jüdischer Sicht mit Konnex zur Bibel, nämlich als Begleittier (näheres dazu in Hoffmann, 2021).

An dieser Stelle sei auch erwähnt, dass die Form der Sternbilder (Linienzug) variieren kann und je nach Kultur unterschiedlich aussieht. Das Bild 2 illustriert die Thematik: für ein Sternbild gibt es unterschiedliche Darstellungsformen. In diesem Bild werden zwei unterschiedliche Muster aus Stellarium verwendet, *Modern* und *Modern (IAU)*. Es gibt entsprechend eine Vielzahl unterschiedlicher Varianten, was natürlich damit zusammen hängt, dass die Sternbilder nicht über ihre Sterne durch die IAU bestimmt sind.

Abschließend noch ein Hinweis auf die **Dark Constellations**. Diese Sternbilder beziehen sich auf die südliche Hemisphäre und entstanden in Australien, Ozeanien, Südamerika oder in den

Abbildung 2: Diese Darstellung ist der Open Source Software Stellarium entnommen. Das Bild zeigt zwei Varianten der Giraffe.

südlichen Regionen von Afrika. Diese Sternbilder sind aber nicht mit den typischen Asterismen vergleichbar. Südlich des Äquators ist die Milchstraße bzw. sind die Staubwolken der Milchstraße besser sichtbar, sodass Figuren oder Tiere in den Staub- und Nebelwolken der Milchstraße erkannt wurden, nicht mehr nur auf Grund von einzelnen Sterngruppen. Für diese Sternbilder gilt aber analog, dass sie in aller Regel als Tiere definiert wurden. Die Definition in den Strukturen der Milchstraße ist auch aus dem mythologischen Standpunkt der jeweiligen Kultur zu verstehen. Beispielsweise interpretierten die Inka in Südamerika den nächtlich über den Himmel wandernden Kosmos als Fluss im Kosmos, der Abend für Abend über ihren Köpfen fließt (Gullberg et al., 2020). In Australien wurde beispielsweise ein Emu von den Aborginies in den Sternen und dem Staub der Milchstraße erkannt und definiert. In anderen Erdteilen wurden, entsprechend der hiesigen Fauna, andere Tiere gesehen: in Südamerika beispielsweise ein Lama. Die **Dark Constellations** sind weniger populär als die von der IAU definierten und in der Bevölkerung zu einem guten Teil bekannten Sternbilder klassischer Art. Nichts desto Trotz ist die historische Perspektive und mythologische Bedeutung der früheren Kulturen in der südlichen Hemisphäre ein archäo-astronomisches Forschungsgebiet mit viel Potential (Hoffmann, 2021). Für diese Arbeit finden diese Sternbilder allerdings keine Berücksichtigung.

Am Ende dieser kurzen Einführung sei noch festgehalten, dass die Verortung am Himmel mittels Sternbilder auch informativ in der heutigen Zeit noch sinnvoll ist, nämlich wenn man bestimmte Ereignisse oder relevante astronomische Objekte einer Gegend zuordnen möchte. Um auf die Giraffe zurück zu kommen: 1788 hat der Astronom Wiliam Herschel eine Spiralgalaxie

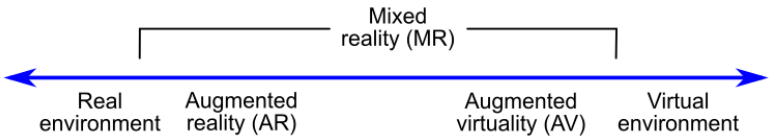

Abbildung 3: Kontinuum gemäß Milgram et al. (1994), die Hololens und das MR-Konzept befinden sich entsprechend der Einteilung auf der linken Seite der Darstellung.

innerhalb dieses Sternbilds entdeckt, welche nicht nur ein ansehnlicher Verteter dieses Galaxientypus ist[1], sondern auch eine 2004 nachgewiesene Supernova enthält (SN 2004dj).

2.2. Mixed Reality

Der Begriff MR bezeichnet Systeme, die die reale Ansicht des Anwenders mit künstlich eingeblendeten Objekten ergänzen und eine Interaktion ermöglichen. Der Terminus Mixed Reality ist aber relativ neu und wurde von Microsoft selbst im Jahr 2016 im Zuge der Einführung der Hololens definiert.

Grundsätzlich sind im historischen Kontext verschiedene Systeme zu unterscheiden: Augmented Reality (AR), Virtual Reality (VR) und neue Techniken wie MR. Für weiterführende Anwendungen bzw. als Überbegriff hat sich der Term Extended Reality (XR) etabliert.

Die ursprüngliche Idee für die Darstellung von digitalen Inhalten in einer Brille oder ähnlichen Systemen geht auf die Arbeit von Sutherland (1968) zurück.

Im Jahre 1994 wurde durch die Arbeit von Milgram et al. (1994) eine Einteilung eingeführt, die sich bis heute im wesentlichen nicht verändert hat. Gemäß dieser Definition liegt das Spektrum der Geräte zwischen der realen Welt und der virtuellen Welt im Sinne von Bild 3. Der Unterschied besteht darin, dass bei AR - Systemen das reale Bild der Natur durch die Brille zu sehen ist und zusätzliche Informationen eingeblendet werden. Am anderen Ende des Kontinuums zeigen Brillen mit VR eine digitale Welt ohne die reale Welt zu berücksichtigen. Man sieht also nur ein künstliches Bild in einer Brille, die keine Informationen von außerhalb zulässt. Diese Techniken sind heute weit verbreitet, beispielsweise durch Spielekonsolen oder vor allem auch als Softwareanwendung in Smartphones, bei welcher die Kamera und die Sensorik mit einem bestimmten Bild kombiniert werden kann. Im Bereich der Geoinformation wird dies beispielsweise gerne für Anwendungen genutzt, die das Bergpanorama mit Gipfeln und der Höhe der Berge illustrieren.

Mit der Definition der MR wurde dieses Kontinuum dahingehend erweitert, dass mit den generierten Objekten, die im Sinne der Agumented Reality dargestellt werden, auf bestimmte Weisen interagiert werden kann. Das bedeutet, dass die Objekte nicht nur in das Sichtfeld einblendet werden, sondern dass diese Hologramme manipulierbar sind und über besondere Features, wie das Verankern des Objektes, eine räumliche Erfahrung erzeugt wird. Für diese Zwecke ist eine komplexe Sensorik notwendig, die Handlungen des Nutzers erkennt. Dies können eye-Tracking Systeme sein, Spracheingaben oder Gesten. Augmented Reality - Brillen wie man sie allgemein kennt, werden oftmals durch eine Hand-Steuerung (eine Art Joystick) ergänzt, der Bewegungen

[1] https://apod.nasa.gov/apod/ap150327.html

Typ / Eigenschaft	Augmented Reality	Virtual Reality	Mixed Reality
Definition	Reale Umgebung wird erweitert	Virtuelle 360°-Umgebung	Kombination aus Realität und virtuellen Objekten
Verhältnis von realer und virtueller Welt	Kombination aus realer Welt und virtuell	rein virtuelle Darstellung	reale und virtuelle Welt sind miteinader verbunden / interaktiv
Interaktionsmöglichkeit	gering	keine	Interaktiv
Gerätetypus	AR-Brillen, Smartphones	VR-Brillen, zB Oculus	MR-Brille, zB Hololens

Tabelle 1: Die Tabelle listet Unterschiede zwischen den jeweiligen Technologien auf.

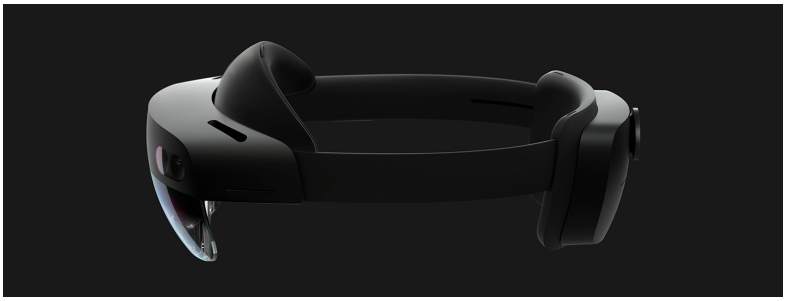

Abbildung 4: Seitliche Ansicht der Hololens. Quelle: Microsoft

des Anwenders und Eingaben registrieren kann. Im Falle der Hololens ist dies nicht nötig, da ein komplexes Zusammenspiel diverser Kameras und Sensoren dies erledigt (siehe Tabelle 17). Die Hololens erfasst durch Kameras und Sensoren die Gesten des Anwenders und ersetzt dadurch den Joystick. Die Unterschiede zwischen diesen drei Technologien sind in Tabelle 1 angeführt.

2.3. Microsoft Hololens

Die Hololens vom Unternehmen Microsoft wurde in Version 1 im Jahr 2015 vorgestellt und war in den USA ab dem Frühjahr 2016 käuflich zu erwerben. Das Unternehmen beschreibt die Hololens wie folgt (Microsoft, 2023b):

> „Microsoft HoloLens ist ein völlig eigenständiger, kabelloser, holografischer Computer. Sie ermöglicht Anwender*innen einen Blick in die sogenannte „gemischte Realität“ (Mixed Reality). Das bedeutet, dass Menschen mit Microsoft HoloLens digitale Inhalte – sogenannte Hologramme – in der realen Welt sehen und mit diesen über Sprache und Gesten interagieren können.“

Für diese Arbeit wird die aktuelle Version der Hololens, die Hololens 2 (HL2), verwendet. Diese ist gegenüber der ersten Ausgabe technisch verbessert und auf Grund der leistungsfähigeren Hardwarekomponenten besser geeignet. Die Hololens ist in Bild 4 dargestellt. Die Version 2 wurde im Jahr 2019 präsentiert und ist mit den Hardwarekomponenten gemäß Tabelle 17 ausgestattet. Bei genauerem Lesen der Ausstattung fällt auf, dass die Hololens über zahlreiche Sensoren und Kameras verfügt, jedoch kein GNSS - Empfänger integriert ist.

Die in der Hololens dargestellten Objekte - Hologramme - werden von Microsoft selbst wie folgt beschrieben (Microsoft, 2023b):

> ”Hologramme sind, einfach gesagt, mit Licht erstellte Grafiken, die dreidimensional im Raum zu schweben scheinen. Wer schon einmal auf einer Veranstaltung mit

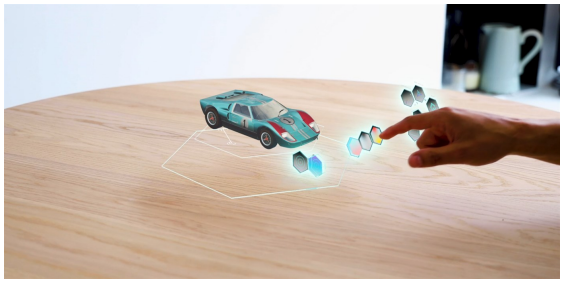

Abbildung 5: Beispielhafte Darstellung der Demoanwendung Ford GT40, Quelle: Microsoft

> Lichtprojektionen war, kennt das wahrscheinlich: Über Laserstrahlen werden Bilder in den freien Raum projiziert. Microsoft HoloLens funktioniert ähnlich, stellt die Bilder aber auf den Projektionsgläsern des Geräts dar, so dass sie erstmals nur die Träger*innen selbst sehen können."

Die Hololens ist also eine Datenbrille, die das Konzept Mixed Reality mit diversen Interaktionsmöglichkeiten umsetzt. Ein Beispiel: für die Hololens gibt es Demoanwendungen, unter anderem jene zum Thema Fort GT40, siehe Darstellung mit Bild 5. Diese Anwendung projiziert das Auto in das aktuelle Sichtbild und man wird aufgefordert, einen Anker zur Orientierung zu setzen. Dies kann zum Beispiel die Schreibtischkante sein. Dann kann man mit den Händen oder anderen Gesten das Auto steuern (drehen, färben, öffnen der Motorhaube, uvm). Durch das Verankern der Position lässt sich mit der Hololens zum Beispiel auch unter das Auto blicken, da die Position nicht verändert wird, das Auto quasi in der Luft festklebt. Diese Demo zeigt, was mit der Hololens möglich ist und wie das Konzept der MR funktioniert. Näheres zu den aktuellen Forschungsbereichen dazu im Abschnitt 3.

3. Forschungsstand

Für dieses Kapitel wurde eine Literaturrecherche durchgeführt. Die Begriffe, wie im Abschnitt Mixed Reality dargestellt, werden oftmals parallel für Untersuchungen verwendet. Übergeordnet hat sich im Lauf der Zeit der Sammelbegriff „Extended Reality“ (XR) etabliert (Çöltekin et al., 2020). Die Anwendungsmöglichkeiten der Hololens sind grundsätzlich vielfältig. Die Hololens ist für Anwendungen gedacht, die Objekte in unmittelbarer Nähe des Anwenders, also zum Greifen nahe, darstellen. Dies ist jedoch nicht zwingend nötig, wie die in diesem Abschnitt dargelegten Veröffentlichungen zeigen. Im nachfolgenden wird auf die Forschungsbereiche Geoinformation und Kartographie, Medizin, Industrie, Unterhaltung, Militär bzw. Verteidigung sowie Astronomie eingegangen.

Auf dem Gebiet der **Geoinformation** sowie der **Kartographie** gibt es eine Vielzahl von Arbeiten, die sich mit der Visualisierung und der Analyse von geographischen Informationen beschäftigen. Zu diesem Bereich zählen für diesen Literaturüberblick auch die Gebiete der Raumordnung und Raumplanung. Die Visualisierung von Geländeformen und Stadtansichten im Sinne einer 3D-Darstellung wird in Wang et al. (2018) beschrieben. Die Autoren definieren Prozesse zu Erstellung und Datenmodelle für den Import in die Anwendung und verweisen in ihrer Zusammenfassung auf die grundlegende Möglichkeit, die Hololens dafür zu verwenden und das die Erfassung von Topographien dadurch leichter fällt, aber auch auf Probleme, die zum Beispiel mit dem eingeschränkten Blickfeld (FoV) zusammenhängen. Ein Szenario für zukünftige Anwendungen ist die Darstellung von Gebäudeplänen (3D-GIS, 3D-Kataster) als MR. Experimentelle Untersuchungen zeigen, dass der Einsatz von MR-Technik mit einer entsprechenden Brille zu besseren Erlebnissen im Vergleich zu herkömmlichen 3D-CAD – Visualisierungen bei den Nutzern führt (Navratil et al., 2020). Auf Grund der Datenmenge und Rechenleistung sind Anpassungen und neue Ansätze zur Verarbeitung von Geoinformationen nötig, die auch in weiterer Folge in XR-Techniken einfließen werden (Guonian Lü and Chen, 2019).

Auf dem Gebiet der **Medizin** dient die Hololens in erster Linie prä-poperativ zur Erkundung der anatomischen Gegebenheiten und zum Vergleich mit anderen Bildgebungsverfahren, die bereits in der Medizin zum Standard gehören (Bitschi et al., 2023). Genauigkeitsuntersuchungen hinsichtlich der Tauglichkeit unterschiedlicher Systeme liefern für die Zukunft beispielsweise neue OP-Techniken, die betroffenen Patienten zu Gute kommen werden (Leger et al., 2017). Insbesondere auch im Bereich der Lehre kann die Hololens zur Darstellung von Körperstrukturen verwendet werden. Die Eignung als Schulungstool für für Aspekte der Orthopädie wurde in Maniam et al. (2020) gezeigt. Generell ist die Mixed Reality Technik in Verbindung mit der Hololens geeignet, um Anatomie zu lehren und die Aufmerksamkeit der Studenten durch die realitätsnahe Darstellung zu erhöhen. Im Zuge dessen kann auch die Anzahl der Sektionen an verstorbenen Personen, eine hohe Qualität der integrierten Objekte in der Anwendung vorausgesetzt, in manchen Fällen ausgelassen werden (Richards, 2020). Die Verwendung von AR- oder VR-Techniken ist in der Medizin schon älter, die Anfänge gehen auf die 90er Jahre zurück (Leger et al., 2017).

Im Bereich der **Industrie** kann die Mixed Reality - Technik zum Beispiel zur Darstellung von Bauteilen oder Konstruktionsplänen verwendet werden. Die im Kapitel 2.3 angeführte Demoanwendung **Fort GT** verdeutlicht dies. Die Technik der MR ist auch verwendbar, um bei der Gebäudewartung verdeckte Leitungen oder komplexe Aufbauten innerhalb von Wänden sicht-

bar zu machen. Dadurch kann das Auffinden von relevanten Strukturen innerhalb von Gebäuden erleichtert werden (Silva et al., 2018). Die Wartung- und Reparaturbedarfserhebung ist durch die XR-Technik auch für größere Anlagen wie Fabriken sinnvoll möglich, wenn eine Verortung und Orientierung des Nutzers verwendet wird. Die Einbettung eines Datensystems über notwendige Aufgaben kann dabei zusätzlich erfolgen (Kunnen et al., 2020). Auch in der Modellierung von Objekt- oder Gebäudestrukturen ist die XR-Techhnik anwendbar. Mit der Arbeit von Bahri et al. (2019) wurde erkannt, dass die Hololens dafür verwendet werden kann, ein building information modelling (BIM) zu erstellen. D.h. die herkömmliche Art und Weise über ein 2D - Modell wird durch die interaktive Variante mit der Hololens ersetzt.

Die **Unterhaltungsindustrie** nutzt Techniken wie AR und VR schon seit langer Zeit für Spieleanwendungen. Manche Spielekonsolen verfügen über eine Erweiterung für eine VR-Brille bzw. sind einfache Datenbrillen mittlerweile auch preislich in einem Bereich angelangt, der für den durchschnittlichen Bürger erschwinglich ist. Im Dunstkreis der Unterhaltung wird die MR-Technik auch für bildungsnahe Verwendungszwecke genutzt. Das Entwickeln von Spielen für Kinder zu Lernzwecken auf Basis der Hololens in Verbindung mit Unity kann für Zwecke der Bildung, in diesem Beispiel in Zusammenhang mit Tieren, eingesetzt werden (Wang et al., 2019). Ein für diese Nutzererfahrungen limitierender Faktor ist das eingeschränkte Sichtfeld über die Hololens. Die Arbeit von Hammady et al. (2019) zeigt, dass über ein optimiertes UI im Verwendungsbereich einer Museumstour die Nutzerfahrung optimiert werden kann. In der Schnittmenge zwischen Unterhaltung und Wissenschaft bzw. Raumplanung kann man aufwendige 3D-Modelle zur Illustration komplexer Entwicklungen, Stichwort SmartCity, nutzen, um das ausgesuchte Areal besser zu visualisieren. Die Arbeit von Zhang et al. (2018) zeigt dies am Beispiel von Toronto. Dies dient nicht nur auf technischer Ebene in der Raum- und Stadtplanung, sondern auch bürgernah zur Darstellung von Planungsprojekten.

Für Zwecke der **militärischen Verwendung** der Hololens gilt wieder, dass die Hololens mannigfaltig verwendet werden kann. Ein Beispiel sind taktische Trainings im Gelände und die zu erwartende Reaktionszeiten. Boyce et al. (2022) schreiben, dass die Umstellung von Tablets oder 2.5D - Visualisierung auf eine interaktive AR-Umgebung in der Hololens nicht zwangsläufig zu Verbesserungen führt. Es wird darauf hingewiesen, dass die ungewohnte Technologie nur dann effizient eingesetzt werden kann, wenn diese gut beherrscht und oft trainiert wird. Für das amerikanische Militär wurde eine spezielle Version der Hololens 2 entwickelt, die unter dem Namen Integrated Visual Augmentation System (IVAS) bei den Truppen getestet wird. Diese angepasste Version soll im Gefechtsfeld mit entsprechenden Informationen zur Entscheidungsfindung beitragen. Jedoch zeigt sich, dass die Verwendung im Gelände zu gesundheitlichen Schwierigkeiten wie Übelkeit oder Kopfschmerzen führt (golem.de, 2023). Auf Grund dieser Probleme und technischen Schwierigkeiten wurde die Einführung um zwei Jahre verschoben (finance.yahoo.com, 2023). Im Bereich der Logistik und Wartungsplanung ist die Mixed Reality Technik eine Möglichkeit, Prozesse effizienter zu gestalten, zeigt Ullo et al. (2019). Eine weitere Möglichkeit betrifft die Missionsplanung für das (theoretische) Schlachtfeld, wo bei der Fokus auf der Visualisierung der örtlichen Gegebenheiten und militärischen Möglichkeiten, zB Schussfeld, liegt (Jenkins et al., 2018). Ein Aspekt, der eigentlich dem medizinischen Bereich zuzurechnen ist, ist die Verwendung von AR- und VR-Techniken bei der Betreuung von verletzten Veteranen mit chronischen Schmerzen. Die Arbeit von Peterson et al. (2021) zeigt, dass die empfundenen Schmerzen während einer Behandlung durch das richtige Einsetzen von AR -

oder VR-Technologien reduziert werden können. Auf Grund der hohen Anzahl von verwundeten Veteranen ist die Thematik ein Teil der Versorgung ehemaliger Soldaten.

Im Bereich der **Astronomie und Astrophysik** findet die Hololens keine nennenswerte Verwendung im Sinne klassischer Forschung. Das Astrophysics data system (ADS) liefert dazu keine relevanten Artikel. Allerdings ist die Technik geeignet, Objekte einprägsam darzustellen und Prozesse aufzuzeigen. Eine entsprechende Anwendung gehört zum Demo-Portfolio von Microsoft und hört auf den Namen **Galaxy Explorer**[2]. Die aufwendig gestaltete App zeigt die Milchstraße, Planten oder die Sonne in einem hohen Detailgrad und ist ein gutes Beispiel dafür, wofür die Hololens genutzt werden kann.

Nach diesem Überblick noch eine Zusammenfassung aus einer Untersuchung über die Forschungsanwendungen der letzten Jahre bzw. seit Einführung der Hololens. In Park et al. (2021) wird dargelegt, dass die Hololens in den vorgenannten Anwendungsbereichen für Forschungsfragen verwendet wird. Weiters wird auf den Bereich der Architektur und Bauingenieurwesen hingewiesen. Die Autoren zeigen, dass sowohl im Bereich der Innenarchitektur wie auch für architektonische Darstellungen im Bereich der Stadtplanung die MR - Verwendung sinnstiftend sein kann. Dass die Nutzererfahrung positiv bewertet wird, zeigt die Arbeit von Xue et al. (2019). In dieser Arbeit wurde erkannt, dass - eingeschränkt auf Teilnehmer aus dem Bereich Aeronautik, Astronautik und Medizin - die Usererfahrung positiv durch Training beeinflusst werden kann. Zusammengefasst lässt sich sagen, dass die Anwendungsbereiche für die MR bzw. XR - Techniken vielfältig sind, aber Aspekte der Nutzung (Training, etc) ein hohes Maß an praktischer Übung und Anwendungserfahrung des Nutzers erfordern, um mit dem Systemen effizient arbeiten zu können.

[2] https://github.com/microsoft/GalaxyExplorer

4. Theoretische Grundlagen

Im nachfolgenden Abschnitt werden die theoretischen bzw. astronomischen Grundlagen behandelt und der Formelsatz für die Umrechnung von äquatorialen Koordinaten in Horizontkoordinaten hergeleitet. Zusätzlich wird die Aktualisierungsgeschwindigkeit (Erdrotation) thematisiert und die Frage einer gegebenenfalls notwendigen kartografischen Projektion erörtert. Der Abschnitt Berechnungsschritte zeigt einen vereinfachten Algorithmus zur Berechnung der erforderlichen Koordinaten für Azimut A und Höhe h. Die hier beschriebenen Grundlagen stellen nur ein Basisfundament dar, eine genaue und detaillierte Behandlung der astronomischen Grundlagen wird in der Fachliteratur (siehe Literaturverzeichnis) besprochen.

4.1. Referenzsysteme

Um Objekte am Himmel oder auf der Erde mit Koordinaten versehen zu können, ist ein entsprechendes Referenzsystem nötig. Unter einem Referenzsystem versteht man ein theoretisches Konstrukt im idealisierten Sinne mit physikalischen bzw. mathematischen Parametern, also sinngemäß wie ein Koordinatensystem definiert wird. Die Umsetzung eines Referenzsystem stellt ein Referenzrahmen dar, der durch Messungen wie zum Beispiel Very Long Baseline Interferometry (VLBI), umgesetzt wird. Man unterscheidet raumfeste Referenzsysteme, also solche Systeme, die idealisiert mit Bezug zu weit entfernten Objekten (z.B. Quasare) als Referenzrahmen gebildet werden. Die zweite Kategorie bilden erdfeste Systeme, die ihre Bezugspunkte auf der Erde haben. Für diese Arbeit beziehen sich Positionsangaben für Sterne immer auf eine bestimmte Epoche, also einem Zeitpunkt, an welchem die Koordinaten bestimmt wurden und in aller Regel auf das System International Celestial Reference System (ICRS), welches auf Basis von Messungen für den Referenzrahmen International Celestial Reference Frame (ICRF) definiert wurde. Das ICRS ist baryzentrisch und an die Parameter des Äquinoktiums für J2000.0 bzw. den Fundamentalkatalog FK5 orientiert (Urban and Seidelmann, 2012). Gegenwärtig ist der Referenzrahmen ICRF in Version 3[3] realisiert. Für erdfeste Bezugssysteme wird das Geozentrum der Erde (Massenzentrum) herangezogen, eine Realisierung für erdfeste Koordinaten stellt das International Terrestrial Reference System (ITRS) dar bzw. die Realisierung als Referenzrahmen (ITRF). Ebenso Erwähnung finden soll an dieser Stelle das WGS84 (World Geodetic System), dass in Zusammenhang mit dem GNSS GPS steht und für Navigationszwecke relevant ist. Die Realisierung erfolgt über ein Referenzellipsoid (Abplattung durch Rotation), einer Geoiddefinition und Fundamentalstationen auf der Erde zur Vermessung. Näheres dazu findet man zum Beispiel in Bertold Witte (2015) und anderen Lehrbüchern der Geodäsie. Im Zusammenhang mit der Definition von Referenzrahmen relevant ist die Bestimmung der Erdrotationsparameter, die international über das International Earth Rotation and Reference Systems Service (IERS) erforscht werden. Die Rotationsparameter fließen in die Bestimmung der Referenzrahmen mit ein und haben einen entsprechenden Einfluss auf die Frage der Zeitmessung (siehe Abschnitt Zeitmessung).

[3] https://hpiers.obspm.fr/icrs-pc/newwww/icrf/index.php

4.2. Koordinatensysteme

Die in dieser Arbeit durchgeführten Berechnungen erfordern ein Basisverständnis von verschiedenen Koordinatensystemen und deren Definitionen. Für die Anwendung in der Hololens ist ein kartesisches, rechtwinkeliges Koordinatensystem erforderlich. Dies ist bestimmt über die normal aufeinander stehenden Achsen für x, y und z wie man es aus der Schulzeit kennt. Ein entsprechender Punkt P in diesem System hat beispielsweise die Koordinaten $P = [1, 3, 4]$. Die Koordinaten für die Sterne im kartesischen System sind das Ende des Prozesses, der Ursprung liegt in sphärischen Koordinaten. In einem sphärischen Koordinatensystem ist der Punkt P auf der Kugel definiert durch drei Angaben: den Radius r vom Mittelpunkt als gerade Strecke zum Punkt, den Polwinkel θ (entspricht dem Winkel vom Pol zum Punkt P) sowie den Azimutwinkel ϕ, also dem Winkel in der x-y Ebene. P kann beispielsweise $P = [2, 37, 68]$ sein. Die Umrechnung zwischen den Systemen bei gegebenen sphärischen Koordinaten kann man standardmäßig in folgender Notation festhalten:

$$\begin{aligned} x &= r \sin\theta \cos\phi \\ y &= r \sin\theta \sin\phi \\ z &= r \cos\theta \end{aligned} \tag{1}$$

Die Umrechnung von kartesischen in sphärische Koordinaten lässt sich mit folgenden Formeln durchführen:

$$\begin{aligned} r &= \sqrt{x^2 + y^2 + z^2} \\ \theta &= \arccos\frac{z}{r} \\ \phi &= atan2(y, x) \end{aligned} \tag{2}$$

Für die Gleichung $\phi = atan2(y, x)$ muss eine Fallunterscheidung durchgeführt werden, die bei der Berechnung am Computer durch die *atan2* - Funktion automatisiert wird. Die Problematik der Fallunterscheidung und die Verwendung der atan2 - Funktion im Code wird später bei der Koordinatenberechnung (siehe 4.7.5) zum Tragen kommen. Die geometrische Veranschaulichung des Punktes P und seiner Parameter ist in Abbildung 6 gezeigt. Die Form dieser Formulierung findet in der Mathematik, Physik und anderen Wissenschaften Verwendung. Die Koordinaten für die Sterne werden als Punkte auf der Himmelskugel gedacht, die in einer unbestimmten Entfernung über dem Beobachter *aufgeblasen* wird. Das Bild 7 verdeutlicht die Überlegung: der Beobachter steht auf dem Punkt O im Mittelpunkt und die Sterne befinden sich auf der Kugeloberfläche mit Radius r.

Für die Positionsangaben auf der Erde, die später in der Hololens bei der Berechnung der Positionen benötigt werden, wählt man folgende Notationen: Für die Orientierung zählt man den Längengrad als Großkreis im Sinne eines Meridians von Nord nach Süd und den Breitengrad parallel zum Äquator (x, y - Ebene) für eine genaue Ortsangabe. Man kann noch die Höhe r dazu nehmen (Radius der Kugel), dann erhält man eine Position im Raum. Für Wien schreibt man zum Beispiel: Längengrad 16,37°, Breitengrad 48,2°. Diese Angabe für sich ist nicht falsch, aber

auch nicht ganz korrekt: die Erde wird zur einfacheren Orientierung in Nord- und Südhalbkugel geteilt, d.h. der Breitengrad von Wien ist +48,2° nördliche Breite beginnend am Äquator. Ein Punkt auf dem selben Meridian (16,37°) südlich des Äquators hätte dann einen Wert zwischen 0 und -90° südliche Breite, also zum Beispiel Kapstadt in Südafrika. Die Breitengradangabe wäre hierfür ca -33.5° südliche Breite. Der Längengrad wird beginnend vom Nullmeridian in Greenwich gemessen und ist eingeteilt in 0 bis 180° östliche Länge bzw. 0 bis -180° westliche Länge; die Angaben für die westliche Länge sind daher immer negativ. Die Grenze bei 180 Grad entspricht der Datumsgrenze.

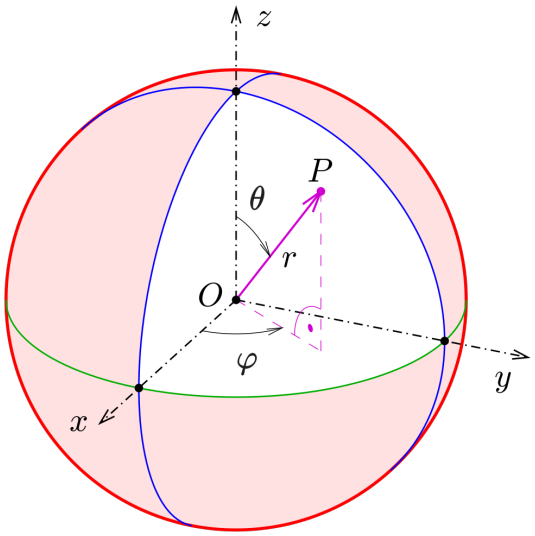

Abbildung 6: Definition eines Punktes P auf der Kugeloberfläche, Quelle: wikipedia.org

In der Astronomie werden in Abhängigkeit des Verwendungszwecks spezifische, dem Bedarf angepasste Koordinatensysteme verwendet. Im wesentlichen definiert man die festgelegten Koordinatensysteme nach ihrem Ursprung wie in Urban and Seidelmann (2012) angeführt:

- topozentrisch - Ursprung auf der Erdoberfläche
- geozentrisch - Ursprung im Mittelpunkt der Erde
- heliozentrisch - Ursprung im Zentrum der Sonne
- baryzentrisch - Massenmittelpunkt des Sonnenystems

Ergänzend dazu kann man noch Systeme definieren, die ihren Ursprung auf dem Mond oder auf anderen Planeten haben. Die Tabelle 2 zeigt die wichtigsten Koordinatensysteme. Zu beachten ist die Ungleiche Händigkeit, Horizont- und ruhendes Äquatorsystem sind linksorientiert, das rotierende Äquatorsystem ist rechtshändig.

Die Idee dahinter ist, sich den Sternenhimmel als große Kugelschale vorzustellen, auf welchen alle Sterne und andere Objekte projiziert sind. Diese Sterne sind auf der Kugeloberfläche

System	Grundkreis	Bezugsmeridian / Nullpunkt	Koordinaten
Horizontal	Horizont	Meridian	Azimut A, Höhe h
Äquatorial, fest	Äquator	Meridian	Stundenwinkel t, Deklination δ
Äquatorial, beweglich	Äquator	Frühlingspunkt	Rektaszension α, Deklination δ
Ekliptikal	Ekliptik	Frühlingspunkt	Ekliptische Länge λ, Ekliptische Breite β
Galaktische Koordinaten	Galaktischer Äquator	Richtung zum galaktischen Zentrum	Galaktische Länge l, Galaktische Breite b

Tabelle 2: Übersicht Koordinatensysteme in der Astronomie

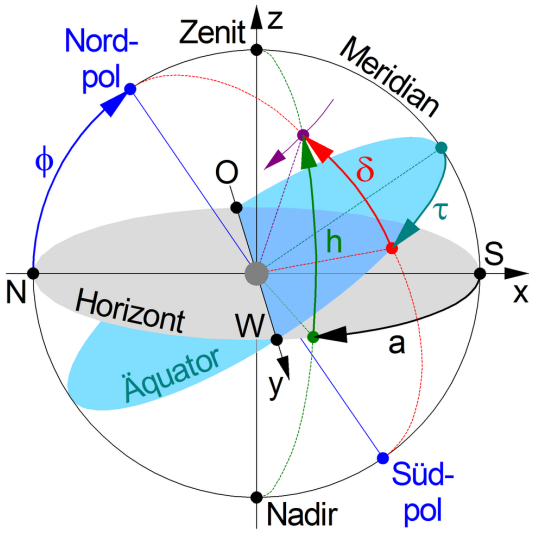

Abbildung 7: Die Kugel zeigt die Lage und Orientierung der Koordinatenpunkte im Horizont- und Äquarorialsystem. Quelle: wikipedia

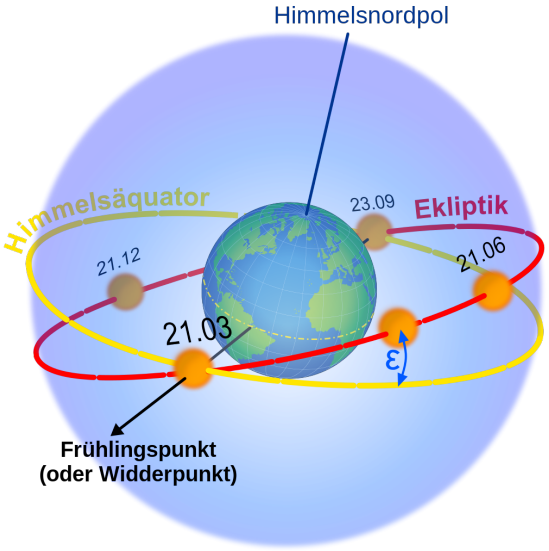

Abbildung 8: Darstellung des Frühlingspunkts / Widderpunktes als Schnittpunkt zwischen Himmelsäquator und Ekliptik. Quelle: wikipedia

liegend gedacht und werden über unterschiedliche Koordinatenformen beschrieben, die aber alle Abwandlungen von sphärischen Koordinaten sind. Zur Orientierung und Beschreibung eignet sich die Darstellung 7. Eine einfache und ursprünglich sinnvoll erscheinende Idee ist das Konzept der Horizontkoordinaten. Der Beobachter sieht von seinem Standpunkt auf der Erde den Horizont vor sich und einen Stern über sich im Sinne der Höhe h über dem Horizont und einem Azimut A, also der Winkel von Norden zum Stern. Das System eignet sich um den Punkt eines Sterns, bezogen auf eine für den Beobachter gültige Uhrzeit, zu beschreiben. Diese Koordinaten werden später auch für die Sternbilder in der Hololens-Anwendung verwendet. Allerdings sind die Angaben nicht orts- und zeitunabhängig. Die Erde rotiert, weshalb sich A ändert und die Höhe h variiert je nach Spur, die der Stern im Laufe der Nacht zieht. Eine koordinative Beschreibung, die sich besser eignet, ist das Äquatorialsystem. Hierbei unterteilt man in das Äquatorialsystem erster Art (fest) und das Äquatorialsystem zweiter Art (rotierend).

Beim festen Äquatorialsystem ist die Höhe eines Sterns, definiert als die Deklination δ, immer stabil, da sich der Bezugspunkt nicht auf den Horizont des Beobachters bezieht, sondern auf den Äquator, der sich nicht ändert. Die zum Azimut äquivalente Angabe ist der Stundenwinkel t, der sich mit festen Bezugspunkt auf den Südmeridian, als Winkel zwischen Stern und Meridian ergibt. Allerdings ist auch der Stundenwinkel orts- und zeitabhängig, weshalb im rotierenden Äquatorialsystem eine Lösung konstruiert wurde, die auch dieses Problem umgeht. Im Äquatorialsystem zweiter Art ist der Winkel für den Drehung definiert als Rektaszension α mit fixem Bezug zum Frühlingspunkt. Der Frühlingspunkt (auch Widderpunkt, Äquinoktium bei Tag- und Nachtgleiche) ist jener Punkt, an dem sich der Himmelsäquator mit der Ekliptik der Sonne schneidet. Der geometrische Zusammenhang wird mit Bild 8 verdeutlicht: die Ekliptik ist ca 23,5° geneigt ($\epsilon = 23,5°$), dort wo sich die Bahn der Ekliptik und der Himmelsäquator treffen, ist der Widderpunkt. Der Zusammenhang zwischen Stundenwinkel und Rektaszension

Koordinatensytem	Ra (h ’ ”)	Dec (° ’ ”)
FK5	06 45 08.91	-16 42 58.03
ICRS	06 45 08.91	-16 42 58.01
Galaktisch	227 13 49.04	-08 53 25.01
Ekliptikal	104 04 54.02	-39 36 18.91

Tabelle 3: Unterschiedliche Koordinatenangaben für den Stern Sirius für die Epoche J2000.0. Die Koordinaten sind sexagesimaler Form gegeben. Quelle: CDS/simbad

ergibt sich aus

$$\alpha = \theta - t \tag{3}$$

Die Rektaszension ermittelt sich also aus der Differenz der Sternzeit minus dem Stundenwinkel. Was es mit der Sternzeit auf sich hat, wird in Abschnitt 4.7.3 erläutert. Wichtig zu beachten ist, dass der Frühlingspunkt kein absolutes Maß im Sinne einer physikalischen Konstante ist, da sich auch dieser Punkt im Lauf der Zeit ändert. Der auch so genannte Widderpunkt (Symbol ♈) lag bei der Definition der Parameter im Sternzeichen Widder, heute liegt er im Sternzeichen Fisch. Daher ist es wichtig, die für astronomische Zwecke benutzte Koordinaten immer auf den Berechnungspunkt (Epoche) zu beziehen (siehe Abschnitt 4.4). In der Praxis werden im Regelfall rotierende äuqatoriale Koordinatenangaben verwendet. Für die Rektaszension erfolgt die Angabe in Stundenwinkel, für die Deklination in Sexagesimalwerten (Grad, Minuten, Sekunden) oder in Dezimalgrad. Ein Beispiel: der theoretisch hellste Stern Sirus hat im System FK5 in der Epoche J2000.0 folgende Koordinaten:

1. Ra 06h 45’ 08.91871”, Dec -16° 42’ 58.0384”
2. Ra 101.28716127°, Dec -16.71612178°

Die Werte in Zeile 1 repräsentieren Koordinaten im Sexagesimalsystem, die Werte in Zeile 2 repräsentieren Dezimalgrad. Abschließend noch ein Beispiel für unterschiedliche Koordinatenangaben in den verschiedenen Systemen für den Stern Sirius mit Tabelle 3 aus dem Katalog SIMBAD Astronomical Database - CDS (Strasbourg) (Simbad).

4.3. Zeit und Ort

Der Anwender betrachtet mit der Hololens die Sternbilder, die lagerichtig dargestellt werden sollen. Dies setzt einerseits die Kenntnis der Position des Anwenders auf der Erde voraus, andererseits die genaue Zeit bzw. das Datum. Nachfolgend ein kurzer Überblick, was man eigentlich unter Zeit im technischen Sinne versteht. Als Orientierung dient die Coordinated Universal Time (UTC), da diese im Alltag für alle am wichtigsten ist. Die zu klärende Frage ist daher, woher man weiß, warum die Giraffe an einem beliebigen Datum zu einer beliebigen Zeit wo am Himmel zu sehen ist.

4.3.1. Die Zeit

Die heute für den Alltag gebräuchliche Definition der Zeit wurde 1972 festgesetzt und als Coordinated Universal Time, abgekürzt mit UTC, bekannt. Diese Uhrzeit ist jene Zeit, die auf Armbanduhren, in Smartphones und anderen Geräten von Menschen genutzt wird, daher spielt sie auch bei der Berechnung der Positionen eine Rolle. Wie diese Zeitform definiert ist und warum sie für die Positionsberechnung relevant ist, wird nachfolgend skizziert.

Die Grundfrage allgemeiner Natur ist, wie man die Zeit als Messung bestimmt. In Verwendung sind unterschiedliche Methoden: atomare Zeitskalen, die dynamische Zeit deren Basis der Weg der Erde um die Sonne ist und Definitionen in Zusammenhang mit der Rotation der Erde sowie die Berücksichtigung relativistischer Effekte. Für den schnellen Überblick über die unterschiedlichen Definitionen und Verwendungen bzw. deren Herkunft dient Tabelle 4.

Bevor man sich damit beschäftigt wie genau man die Zeit misst, braucht man aber ein Normal für eine Zeiteinheit, mit der man rechnen kann. Die Definition einer Sekunde ist historisch gewachsen und wir beginnen bei der aktuell gültigen Definition als Standardeinheit für die Zeitmessung im Sinne der SI-Einheit. Diese wurde 2018 von der Bureau International des Poids et Mesures (BIPM) auf Basis physikalischer Konstanten wie folgt definiert:

> „Die Sekunde, Einheitenzeichen s, ist die SI-Einheit der Zeit. Sie ist definiert, indem für die Cäsiumfrequenz *deltav*, der Frequenz des ungestörten Hyperfeinübergangs des Grundzustands des Cäsiumatoms 133, der Zahlenwert 9 192 631 770 festgelegt wird, ausgedrückt in der Einheit Hz, die gleich $s^{-}1$ ist.“

Historisch gesehen ging dieser Definition die Beschreibung der Sekunden folgendermaßen voraus. Im Jahr 1960 wurde die Sekunde als Ephemeridensekunde wie folgt definiert:

> „The second is the fraction 1/31 556 925.9747 of the tropical year for 1900 January 0 at 12 hours ephemeris time.”

Diese Definition wurde 1967 durch eine technisch-physikalische Lösung abgelöst.

> „The second is the duration of 9 192 631 770 periods of the radiation corresponding to the transition between the two hyperfine levels of the ground state of the caesium 133 atom.”

Die vorgenannte Bestimmung der SI-Einheit Sekunde hat sich bis 2018 gehalten. Dann wurde die technische Definition zwar nicht geändert, aber die Beschreibung angepasst. Bei der Konferenz 2018 wurden die SI-Einheiten grundsätzlich präziser beschrieben und somit auch die Definition der Sekunde adaptiert.

Eine frühe Definition der Sekunde, die lange Gültigkeit hatte, ist die Sonnensekunde. Diese wurde, unter der Annahme, dass die Erdrotation absolut konstant ist, als 1/86400er Teil eines mittleren Sonnentages festgelegt. Es hat sich aber gezeigt, dass die Rotationsgeschwindigkeit der Erde nicht konstant und leicht rückläufig ist, daher wurde 1960 eine neue Definition der Sekunde eingeführt, die Ephemeridensekunde (die Definition ist oberhalb angeführt). Diese basiert nicht mehr auf einer starren Rotation, sondern auf der Bahnbewegung der Erde um die Sonne. Diese Definition wurde 1967 abgelöst durch eine rein physikalische Lösung auf Basis der ersten Atomuhren. Die historische Entwicklung der Definition ist in Tabelle 5 skizziert.

Zurückkommend auf die eingangs erwähnte Unterteilung möglicher Messarten hier ein kurzer Überblick: In der Gruppe der **atomaren Zeitskalen** ist die International Atomic Time (TAI) Basis der Messung. Diese basiert technisch auf dem oben erwähnten Messnormal für die Sekunde, basiert also auf einer physikalischen Konstante des Hyperfeinstrukturübergangs im Cäsium-Atom. Die Terrestrische Zeit (TT) wird durch die IAU definiert. Sie bestimmt sich aus der TAI und wird vom BIPM jährlich veröffentlicht. Die TT erhält man durch Verwendung der TAI und Addition einer Konstante:

$$TT = TAI + 32.184s \tag{4}$$

Die Gleichung ist gültig ab dem 1. Jänner 1977 für 0:00 TAI. Eine genaue Realisierung setzt sich aus der Ergänzung durch das modifizierte Julianische Datum (MJD), einer für das Jahr gültigen Konstante in Nanosekunden sowie einem Korrekturfaktor zusammen.

Die Zeitrechnung unter Zuhilfenahme der **Erdrotation** weicht naturgemäß von der hochpräzisen Messung über Atomuhren ab. In der Gruppe der Messmethoden für rotationsbedingte Zeitmessungen finden sich die UT0, UT1, UT2, GAST, GMST, LAST und LMST. Die UT (Universal Time) ist ein Vertreter dieser Rotationszeitmessung. Der Durchgang durch den Nullmeridian bei Greenwich wird mit Hilfe der mittleren Sonne bestimmt und die Zeit gemessen. Eine verbesserte Variante ist die UT1, die mit Hilfe des Phasenwinkels der Erde als Grundlage für die UTC herangezogen wurde. Die UT2 ergänzt noch die Schwankungen der Erdrotation. Greenwich Apparent Sidereal Time (GAST) und Greenwich Mean Sidereal Time (GMST) stehen für die Sternzeit für den Meridian bei Greenwich. Die mittlere GST wird auch später bei der Berechnung der Koordinaten für die Sterne verwendet. Die Zeitmessung Local Sidereal Time (LAST) und Local Mean Sidereal Time (LMST) beziehen sich auf die örtliche gemessene Zeit, d.h. man addiert zur Greenwich - Zeit die lokale Zeit (Längengrad) und spielt bei der Berechnung ebenfalls eine Rolle (siehe 4.7.3). Die Bezeichnung LAST beschreibt die lokale scheinbare Sternzeit, und der Term LMST steht für die lokale mittlere Sternzeit am jeweiligen Ort.

Der dritte Vertreter ist die **dynamische Zeit**, also die Rotation der Erde um die Sonne und daraus abgeleiteter Werte. Dazu zählt die früher gebräuchliche Ephemeridenzeit sowie die TDT bzw TDB. Zurückkommend zur UTC als für den Alltag relevante Zeit gelten folgende Zusammenhänge: Die UTC wird aus der TAI abgeleitet und um Schaltsekunden ergänzt, da die Rotation der Erde nicht konstant ist. Die Ungleichheit wird stets < 1 Sekunde gehalten. Da die UTC als atomare Zeitskala im Vergleich zur UT stets etwas schwankt, werden die Korrekturen periodisch nachgereicht.

$$UTC = TAI - Schaltsekunden \tag{5}$$

Abschließend noch ein Blick auf die zeitlichen Unterschiede. Im wesentlichen lässt sich sagen, dass für eine beliebige Zeit MEZ die Abweichung zwischen UTC und UT im Millisekundenbereich liegt, die Abweichung zur TAI die bereits erwähnten 37 Sekunden beträgt und die Differenz zwischen UTC und TT nochmals etwa 42 Sekunden veranschlagt. Für diese Arbeit nicht relevant und auch rechnerisch nicht berücksichtigt sind weitere Definitionen, die ihren Ursprung in der Berechnung von Ephemeriden haben. Hiezu zählen die geozentrische Koordinatenzeit (TCG), die baryzentrische Koordinatenzeit (TCB) und die baryzentrische dynamische Zeit TDB. Die TT ist auch Teil dieses Ensembles und wurde oben bereits erwähnt. Diese vier

Name	Abkürzung	Bezug
Internationale Atomzeit	TAI	atomar
Ephemeridenzeit und dynamische Zeit	ET	Revolution
Weltzeit	UT	Rotation
Koordinierte Weltzeit	UTC	atomar
Sternzeit	SD	Rotation

Tabelle 4: Auflistung der teilweise historischen und aktuellen Zeitdefinitionen und deren Verwendung

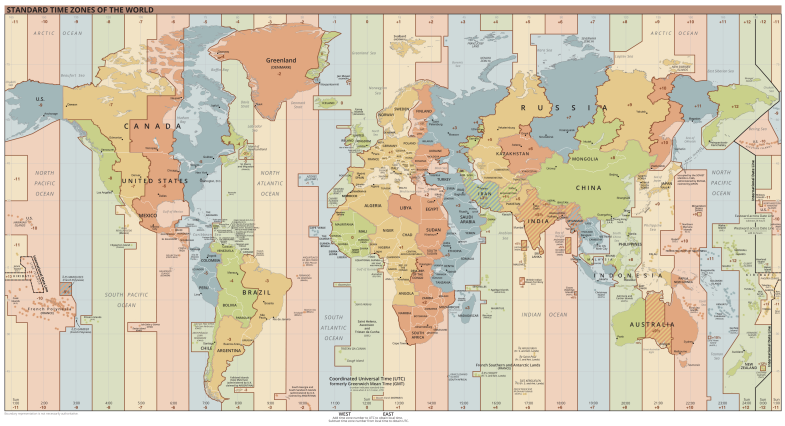

Abbildung 9: Zeitzonendarstellung, Quelle: wikipedia.org

Definition fasst man unter dem Begriff Dynamische Zeit zusammen. Bei diesen Zeitdefinitionen werden relativistische Effekte berücksichtigt, entsprechend komplex ist die Definition und Berechnung.

Das **Datum** ist deswegen von Bedeutung, weil die Erde auf ihrem Weg um die Sonne die Lage der Sternbilder entsprechend dem Kalenderstand ”verschiebt”, d.h. im März ist die Giraffe anders am Himmel positioniert als im Oktober. Die Schiefe der Erdbahn (näherungsweise 23,5 Grad) und der elliptische Umlauf um die Sonne sorgen nicht nur für die Jahreszeiten, sondern auch für das unterschiedliche Erscheinungsbild des Nachthimmels. Es ist also nicht nur die reine Tageszeit für die Orientierung nötig, sondern auch der Ort der Erde auf seiner Umlaufbahn um die Sonne. Historische Details zur Entstehung der Kalender sind in Vogtherr (2012) dargestellt, die Bestimmung und Berechnung der Zeit ist beispielsweise in Urban and Seidelmann (2012) oder Hanslmeier (2020) nachzulesen.

4.3.2. Der Ort des Beobachters

Es ist naheliegend, dass der Standort des Beobachters für die Visualisierung von entscheidender Bedeutung ist. Beispielsweise ist die Position eines Sterns am Himmel naturgemäß abhängig vom Breitengrad, auf welchem sich der Beobachter auf der Erde befindet.

Betrachtet man über einen längeren Zeitraum die Verlaufsspuren von Sternen mit Mittelpunkt

Definition der Sekunde	Grundlage	Nutzung von - bis
Sonnensekunde	Erdrotation	bis 1960
Ephemeridensekunde	Erdbahn	1960 bis 1967
Atomsekunde	Atomuhr	seit 1967

Tabelle 5: Das Messnormal für die Definition der Sekunden im Laufe der Zeit.

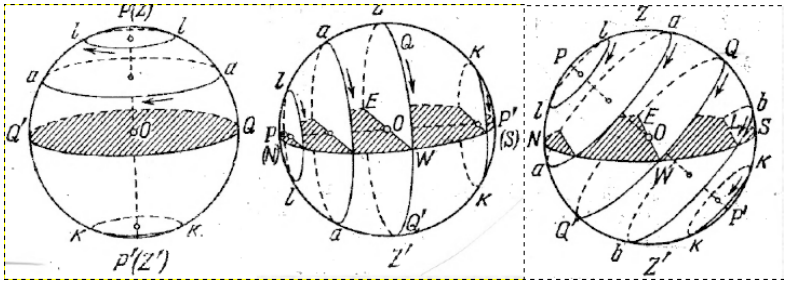

Abbildung 10: Die Grafik zeigt die unterschiedliche Darstellung der Sternspuren am Nachthimmel. Das erste Bild links zeigt den Verlauf vom Standpunkt Nord(pol), das zweite Bild zeigt den Verlauf vom Äquator aus gesehen. Das Bild rechts entspricht dem Blick in den Nachthimmel in der nördlichen Hemisphäre (Österreich, Deutschland).

nördlicher Polarstern, entstehen Bilder wie mit Abbildung 11 gezeigt. Wie sich die Spuren der Sterne präsentieren, hängt aber von der Position ab: am Nordpol ist der Verlauf gänzlich anders als am Äquator, und auf der südlichen Hemisphäre ist die Drehrichtung entgegengesetzt. Die Zusammenstellung mit Bild 10 zeigt die unterschiedlichen Verlaufskurven. Die Länge bzw. der Längengrad ist ebenfalls zu berücksichtigen, da dies für die Berechnung der Zeit bzw. der Sternzeit relevant ist.

Für die korrekte Darstellung der Sternbildpositionen bzw. der einzelnen Sternörter ist also die Kenntnis der Position des Anwenders auf der Erde nötig. Die koordinative Position findet daher entsprechend Niederschlag in den Umrechnungsformeln für die Koordinaten eines Sterns zu einem bestimmten Zeitpunkt, siehe Abschnitt 4.7.5.

4.4. Sternpositionen mit hoher Genauigkeit

In der Astronomie bzw. dem Fachbereich der Astrometrie (aus dem griechischen für Stern und Maß, Messen) würde man für die Position eines Sterns komplexe Berechnungen berücksichtigen, die für diese Arbeit auf Grund der nur eingeschränkt genauen Positionierung in der Hololens-Darstellung nicht relevant sind. Formal zu berücksichtigen sind die folgenden Effekte, die nicht erschöpfend angeführt sein wollen:

- Eigenbewegung der Sterne
- Präzession und Nutation

Abbildung 11: Sternspuren des Nachthimmels in Deutschland. Die Mitte zeigt den Polarstern, um welchen die anderen Sterne kreisen. Mit freundlicher Genehmigung von Michael Luy.

- Parallaxe und Aberration
- Atmosphärische Effekte
- Topozentrische Koordinaten

Die obige Aufzählung illustriert die Komplexität der genauen Positionsbestimmung und verschleiert die aufwendige Mathematik dahinter auf elegante Weise. Eine genaue Beschreibung eines (mittleren) Sternortes würde folgender Formel aus Urban and Seidelmann (2012), Gleichung 7.125, genügen:

$$u_4(t') = R_3(-\beta)R_\epsilon f[g[u_b(t_0) + (t - t_0)u_b(t_0) - E_B(t)]] \tag{6}$$

Die Variablen in dieser Formel seien hier kurz angeführt:
Die scheinbare Position eines Sterns $u_4(t')$ ist gegeben durch

- t' Beobachtungsepoche in TT-Zeit
- t Beobachtungsepoche in TBT-Zeit
- t_0 Referenzepoche Sternkatalog, TBT-Zeitskala
- $u_B(t_0)$ Katalogposition des Sterns
- $u_b(t_0)$ Bewegungsvektor des Sterns (Eigenbewegung, Parallaxe, Radialgeschwindigkeit)
- E_B Baryzentrische Position der Erde
- $g[]$ Funktion für die gravitative Ablenkung des Lichts

- $f[]$ Funktion für Abberation
- R_σ Transformation der Koordinaten
- $R_3(-\beta)$ Rotation

Details zu diesen Variablen sind in Urban and Seidelmann (2012) erläuert.

Im folgenden wird ein kurzer Überblick mit groben Werten für Korrekturparameter gegeben, eine genauere mathematische und physikalische Betrachtung der Effekte würde aber den Rahmen dieser Arbeit überdehnen. Die mathematische und physikalische Beschreibung ist aus den Werken Urban and Seidelmann (2012), Montenbruck (2005), Richter (2005) und Karttunen (2005) entnommen, in den jeweiligen Abschnitten wird ergänzend darauf verwiesen.

4.4.1. Eigenbewegung der Sterne

Die **Eigenbewegung der Sterne** bezieht sich auf die Veränderung des jeweiligen Sterns auf der als feststehend gedachten Position auf der Himmelskugel. Dass sich die Sterne bewegen und keine raumfesten Koordinaten haben, geht auf Untersuchungen von Edmund Halley zurück. Die Eigenbewegung beinhaltet eine Positionsveränderung für die Rektaszension (Ra) sowie für die Deklination (Dec). Ergänzend dazu muss man die Radialgeschwindigkeit v_r eines Sterns miteinbeziehen, die sich darauf bezieht, ob sich ein Stern auf die Erde zu oder weg bewegt (Beobachtungen zum Dopplereffekt, der für die Bestimmung der Rot- oder Blauverschiebung benutzt wird). Typische Werte für die Eigenbewegung betragen 0.1" pro Jahr. Jener Stern, Stand dieser Arbeit, mit der größten Eigenbewegung ist der Stern *Barnard* (im Katalog HIP mit Nr. 87937) mit einer Eigenbewegung der $\mu_\alpha = -800.5 mas/y$ zu $\mu_\delta = 10.360 mas/y$ (Datenlage Gaia DR3, Stand 2022). Die Eigenbewegung ergibt sich nach Karttunen (2005) aus

$$\mu = \sqrt{\mu_\alpha^2 * \cos^2 \delta + \mu_\delta^2} \tag{7}$$

wobei der Winkel cos bei 0 liegt, wenn die Deklination unverändert bliebe. Dieser Term kann als lineare Abschätzung verstanden werden und lässt sich ausdrücken als $\alpha_{corr} = \alpha + \tau \mu_{alpha}$, wobei τ für das Datum (Jahresanteil) steht, siehe Urban and Seidelmann (2012); analog die Formulierung für die Dec. Die allgemeine Formulierung der Position eines Sterns, gegeben als Positionsvektor r mit $\alpha =$ Ra, $\delta =$ Dec und $\mathbf{r}$ als Entfernungsvektor, der über die Parallaxe bestimmt wird, lautet

$$\mathbf{r} = \begin{matrix} r \cos\delta \cos\alpha \\ r \cos\delta \sin\alpha \\ r \sin\delta \end{matrix} \tag{8}$$

und kann nach Ableitung d/dt als Matrix $\dot{r}$ mit den Termen der Eigenbewegungen ausgedrückt werden um die Position zum aktuellen Zeitpunkt mit Referenz zu einer Epoche, zu bestimmen.

Diese schleichende Bewegung der Sterne, die sich nur über einen längeren Zeitraum gut erfassen lässt, ist mit ein Grund für die Veränderlichkeit von Sternbildern. Dieser Aspekt ist aber nur für größere Betrachtungszeiträume und bei historischen Analysen relevant. Möchte man wissen, wie das Sternbild der Giraffe vor 3000 Jahren ausgesehen hat oder es in 2000 Jahren

aussehen wird (auf Basis der aktuell verfügbaren Kenntnisse der Sternbewegungen), wäre die Eigenbewegung der Sterne ein relevanter Berechnungsfaktor.

4.4.2. Präzession und Nutation

Die Präzession und die Nutation sind Bewegungen der Erde, die beide miteinander verknüpft sind. Bei der Präzession handelt es sich um eine Kreiselbewegung der Erdachse, die sich auf Grund von gravitativen Einflüssen auf die Erde ergibt. Diese Bewegung ist, genau betrachtet, eine elliptische Bahn die für einen Umlauf etwa 25 700 Jahre benötigt. Streng genommen müsste man die Präzession aufsplitten in einen Anteil für die Präzession der Ekliptik und einen äquatorialen Teil (Urban and Seidelmann, 2012). Auf dieser Bahn vollzieht die Erde aber noch eine zusätzliche Bewegung, die sogenannte Nutation. Hierbei handelt es sich um eine Kurve ähnlich einer Sinus- oder Cosinus-Funktion, die Amplitude hat eine Dauer von ca 19 Jahren. Eine Darstellung dieser beiden Bewegungen zeigt das Bild 12. Mathematisch betrachtet werden diese beiden Bewegungen über Rotationsmatrizen berücksichtigt. Diese Bewegung ist auch der Grund, warum bei Sternpositionen stets ein bestimmtes Datum (Epoche) angegeben werden muss und für Positionsfragen zu einem bestimmten Datum eine Korrektur anzubringen ist (Unsöld, 2005). Die Verschiebung des Äquinoktiums auf der Ekliptik beträgt jährlich etwa 50.3". Die Herleitung und die Konstanten für die Winkel wurden von der IAU 1976 definiert (Karttunen, 2005).

Die mathematische Formulierung beider Effekte ist komplex, in Lehrbüchern wird oft auf die Grundlagen bzw. Notation von Lieske et al. (1977) verwiesen. Eine verallgemeinerte Formulierung der Präzession lässt sich schreiben als

$$P_0 = \Upsilon R - \Upsilon_0 R \tag{9}$$

Zur Illustration dient eine Zusammenfassung aus Karttunen (2005), welche die Rechenschritte in kurzen Blöcken aufzeigt: genutzt wird der Richtungsvektor des Sterns in Matrixschreibweise analog zum vorigen Abschnitt für die verwendete Epoche (in aller Regel J2000.0), daraus resultiert die Matrix P_0. Die entsprechenden Matrizen für die Rotation werden mit P_0 multipliziert, sodass der Vektor auf die aktuelle Position zeigt.

Zur Berechnung der Präzessionsmatrix sind 3 Winkel und das julianische Datum bzw. t nötig. Den Faktor t erhalten wir aus obiger Herleitung, die Winkel η, ζ, θ sind wie folgt definiert:

$$\begin{aligned} \eta &= 2306.2181 * t + 0.30188 * t^2 + 0.01799 * t^3 \\ \zeta &= 2306.2181 * t + 1.09468 * t^2 + 0.018203 * t^3 \\ \theta &= 2004.3109 * t - 0.42665 * t^2 - 0.041833 * t^3 \end{aligned} \tag{10}$$

Anm: In Urban and Seidelmann (2012) wird dargelegt, dass die drei Parameter η, ζ, θ nur für Berechnungen für Daten vor 2006 gültig sind, die Tabelle der Koeffizienten wurde aktualisiert und ist in Hilton et al. (2006) mit aktuellen Werten aufgelistet. Ausgewertet mit den Koeffizien-

ten ergibt sich folgende Matrix

$$P = \begin{pmatrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_7 & P_{23} \\ P_{31} & P_{32} & P_{33} \end{pmatrix} \tag{11}$$

und die zugehörigen Gleichungen:

$$\begin{aligned} P_{11} &= \cos\zeta\cos\theta\cos\eta - \sin\zeta\sin\eta \\ P_{12} &= -\cos\zeta\cos\theta\sin\eta - \sin\zeta\cos\eta \\ P_{13} &= -\cos\zeta\sin\theta \\ P_{21} &= \sin\zeta\cos\theta\cos\eta + \cos\zeta\sin\eta \\ P_{22} &= -\sin\zeta\cos\theta\sin\eta + \cos\zeta\cos\eta \\ P_{23} &= -\sin\zeta\sin\theta \\ P_{31} &= \sin\theta\cos\eta \\ P_{32} &= -\sin\theta\sin\eta \\ P_{33} &= \cos\theta \end{aligned} \tag{12}$$

Abschließend erhält man aus dem Produkt $P_0 * P$ die entsprechenden Koordinaten für den jeweiligen Zeitpunkt.

Die obige Darstellung entspricht der Berechnung für größere Ansprüche an die Genauigkeit. Weiterführende Erläuterungen und Verweise auf entsprechende Literatur (insbesondere Festlegungen der IAU) findet man in Urban and Seidelmann (2012). Einfache Näherungsformeln für die grobe Abschätzung, die auch in der Praxis Anwendung findet, stellen die beiden folgenden Gleichungen aus Meeus (1998) dar:

$$\begin{aligned} \triangle\alpha &= m + n\sin(\alpha)\tan(\delta) \\ \triangle\delta &= n\cos(\alpha) \end{aligned} \tag{13}$$

Die vorgenannten Näherungsformeln ergeben sich durch Ableitung aus den Umrechnungsformeln für die Transformation von äquatorialen Koordinaten und der Ekliptik, die Herleitung ist in Karttunen (2005) erklärt. Die Werte m sowie n sind gegeben in Sekunden durch $m = 3.07496 + 0.00186T$ bzw $n = 1.33621 - 0.00057T$, wobei T gleich der Jahrhunderte seit der Epoche 2000.0 ist. Zwar sind m und n nicht absolut konstant, aber durch die sehr geringe Änderung werden sie als Präzessionskonstanten aufgefasst. Änderungen in den Werten erfolgen für viele Jahre erst ab der 3. oder 4. Kommastelle, entsprechende Tabellen sind in den vorhin erwähnten Werken angeführt. Erst bei sehr langen Zeiträumen fallen die Änderungen theoretisch ins Gewicht, so benötigt beispielsweise m einen Zeitraum von 500 Jahren um von 3.069s auf 3.079s anzusteigen.

Die Berechnung der Nutation ist ebenfalls komplex und soll für diese Arbeit nur in Form von Näherungsformeln angegeben werden. Die Nutation in der Grafik 12 wird dargestellt in

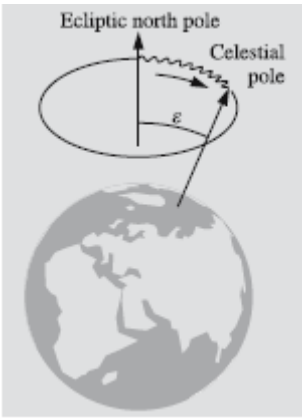

Abbildung 12: Die Grafik zeigt die Präzessionsbewegung der Erde und das Schwanken durch die Nutation. Quelle: Fundamental Astronomy

Form einer gewissen Amplitude („eiern “, to wobble im englischen), die sich aber nur auf den ekliptischen Längengrad bzw. die Schiefe auswirkt. Die Nutationsmatrix kann gemäß Emerson (1973) geschrieben werden als $N = R_1(-\epsilon)R_3(-\triangle\psi)R_1(+\epsilon_A)$. Eine weniger präzise Berechnung beruht auf einer Vereinfachung, aus welcher sich folgende Korrekturformeln für Rektaszension und Deklination ergeben:

$$\begin{aligned} \triangle\alpha &= (\cos\epsilon + \sin\epsilon \sin\alpha \tan\delta)\triangle\psi - \cos\alpha \tan\delta\triangle\epsilon \\ \triangle\delta &= \sin\epsilon \cos\alpha\triangle\phi + \sin\alpha\triangle\epsilon \end{aligned} \tag{14}$$

Gemäß Meeus (1998) sind die Gleichungen für die Nutation, also die Näherungsformeln, für Sterne nahe den Polen nicht verwendbar. Die anzubringenden Korrekturwerte sind klein, daher auch im Kontext der Arbeit nicht relevant.

4.4.3. Parallaxe und Aberration

Zur Veranschaulichung der Parallaxe dient folgendes Bild: beobachtet man einen Stern zum Zeitpunkt t_0 und denkt sich die Linie zum Stern als Gerade, so bedingen die Rotation der Erde, die Bewegung der Erde um die Sonne und die Bewegung des Sonnensystems eine Verschiebung dieser Geraden. Der entstehende Parallaxenwinkel zum Zeitpunkt t_1 wird eingeteilt die tägliche Parallaxe (Erdrotation), jährliche Parallaxe (Bewegung um die Sonne) und die sakuläre Parallaxe, womit die Bewegung des Sonnensystems relativ zum Fixtsternhimmel gemeint ist.

Die Aberration bezeichnet die Ablenkung des Lichts der Sterne. Grund dafür sind relativistische Effekte (Endlichkeit der Lichtgeschwindigkeit ν) und die Bewegung der Erde im Raum. Diese Bewegungen beginnen bei der Rotation, der Bahn der Erde um die Sonne, die Bewegung der Sonne, der Galaxie und so weiter. Die Laufzeit des Lichts mit einer Geschwindigkeit von $\nu = 299792458 m/s$ führt also dazu, dass die Position des Stern und die des Beobachters nicht konstant bleiben. Für dieses Phänomen wird in der Literatur, zB in Hanslmeier (2020), gerne

folgender Vergleich beschrieben: man denke sich senkrecht einfallenden Regen und eine Person, die mit dem Regenschirm stehend sich vor dem Regen schützt. Wenn sich diese Person nun (schnell) bewegt, hält sie den Schirm schräg nach vorne, um sich weiter vor dem Regen schützen zu können. Man teilt den Effekt der Aberration: jährliche und tägliche Aberration. Die jährliche Aberration bezieht sich auf die Umlaufgeschwindigkeit der Erde um die Sonne (grob 30km/s) und die tägliche Aberration bezieht sich auf die Rotation der Erde (siehe Abschnitt 4.5). Die Herleitung fußt auf der klassischen Mechanik und der Galilei-Transformation zwischen verschiedenen Bezugssystemen.

4.4.4. Atmosphärische Effekte

Die tatsächliche Lage von Himmelsobjekten im Raum wird durch die Atmosphäre verändert. Diese sogenannte Atmosphärische Refraktion verändert die Höhe über dem Horizont. Allgemein gilt: je näher ein Objekt am Horizont steht, desto stärker ist der Effekt. Am deutlichsten ist dieser Effekt bei der untergehenden Sonne zu bemerken: wenn sie tief am Horizont steht, ist ihre tatsächliche Position schon unterhalb des Horizonts, als Faustregel gilt für tief stehende Objekte eine Verschiebung um 30“nach Karttunen (2005). Die Berechnung des Effekts ist abhängig von unterschiedlichen Parametern der Atmosphäre wie Feuchtigkeit, Temperatur, Druck und der Höhe des Objekts. Es haben sich mehrere Varianten der Berechnung etabliert, die hier kurz angeführt werden sollen. Eine Variante der Berechnung ist die Integration über ein modifiziertes Integral für die herkömmliche Refraktionsintegration, siehe Urban and Seidelmann (2012).

$$\xi = -\int_0^z \frac{rdn/dr}{n + rdn/dr} dz \tag{15}$$

Dieses Integral berücksichtigt bei seiner Lösung unterschiedliche Atmosphärenschichten (Tropopause, Stratosphäre) und weitere Verbesserungen. Die Lösung wird mit konstanten Parametern der Erde gefüttert und entsprechend entwickelt bzw. numerisch gelöst. Die Berechnung ist in Urban and Seidelmann (2012) erläutert. Die Gesamtrefraktion ξ setzt sich durch Addition der Ergebnisse für die Tropopause und die Stratosphäre zusammen, entspricht also

$$\xi_{gesamt} = \xi_{tropo} + \xi_{strato} \tag{16}$$

Der obige Ansatz mittels Integration ist am genauesten, in der Praxis werden oft Vereinfachungen verwendet. Eine davon ist die Formel von Saastamoinen. Eine entsprechende Herleitung dazu ist in Saastamoinen (1972) nachzulesen. Die dafür abgeleitete Formel für die Refraktion ξ stellt sich wie folgt dar:

$$\xi = 16”.271 Q \tan_{z0}(1 + 0.0000394 Q \tan^2(z_0) - 0.0749”(\tan(z_0) + \tan^3(z_0) P(1/1000) \tag{17}$$

Der Term Q steht in der Formel für $(P - 0.156e)/T$. Der üblichen Konvention entsprechend ist P der Druck, T die Temperatur, z_0 die gemessene Zenitdistanz. Eine weiterer, vereinfachter Ansatz zur Bestimmung der Refraktion ist die nachfolgende Herleitung. Grundlage dafür ist im wesentlichen das Brechungsgesetz von Snellius (Hanslmeier, 2020):

$$\frac{\sin \alpha_1}{\sin \alpha_2} = \frac{n_2}{n_1} \tag{18}$$

mit n_1, n_2 für den Brechungsindex des jeweiligen Mediums. Aus dieser Gleichung lässt sich nach Umformungen und für z_b = wahre Zenitdistanz folgende Formel für die Refraktion R ableiten:

$$R = (n - 1) * \tan(z_b) \tag{19}$$

Die Herleitung für $R = (n-1)\tan(z_b)$ ist in Karttunen (2005) illustriert. Daraus ergibt sich die Formulierung wie man sie oft in Publikationen findet:

$$R = \frac{p}{273 + T} 0.00452° \tan(90° - a) \tag{20}$$

Obige Formel ist für Werte über 15° Höhe über dem Horizont anzuwenden, darunter muss die Erdkrümmung berücksichtigt werden. Je nach Literatur finden sich Varianten dieser Formel, zB Urban and Seidelmann (2012) oder Montenbruck (2005). Die Refraktion R wird in Grad gemessen, p ist der Luftdruck, T die Temperatur, a der Höhenwinkel. Für Standardwerke mit entsprechenden Tabellen wie zum Beispiel den *The Astronomical Almanac* werden passende Konstanten für Druck, Temperatur oder Feuchtigkeit verwendet.

Ein weiterer Effekt der Atmosphäre sei hier noch erwähnt: Seeing. Damit ist die Störung gemeint, die sich durch die turbulente Atmosphäre ergibt, begrifflich als Szintillation bekannt. Diese Thematik ist bei astronomischen Beobachtungen mit großen Teleskopen vom Erdboden aus relevant und für die Qualität der Beobachtung von Bedeutung, wird durch adaptive Optiken, optimale Beobachtungsorte und Wetterbeobachtung optimiert. Für diese Arbeit hat dies keine Relevanz.

4.4.5. Topozentrische Koordinaten

Wir rechnen für die sphärischen Koordinaten ausgehend von einem Koordinatensystem mit geozentrischen Mittelpunkt, d.h. der Schnittpunkt aller drei Koordinatenachsen (x, y, z = 0 bzw. $\phi, \lambda = 0$) liegt in der Mitte der Erde. Genauer betrachtet befindet sich der Beobachter aber auf einem Punkt im Sinne von topozentrischen Koordinaten, d.h. man müsste hierfür die Position auf der Kugelschale in einer gewissen Höhe berücksichtigen, also zum Beispiel den Mittelpunkt des Koordinatensystems auf die Höhe bzw. Position von Wien legen. Dies führt zu einer Verschiebung, die rechnerisch berücksichtigt werden muss, wenn hohe Genauigkeit erforderlich ist. Relevant ist dies bei der Beobachtung der Sonne bzw. dem Mond und den Ephemeriden für die Planeten. Im Falle der Sterne wird die Horizontalparallaxe

$$p = \arcsin \frac{\rho}{r} \tag{21}$$

so klein, dass der Wert nicht mehr relevant bzw. messbar ist. Gemäß Montenbruck (2005) beträgt die Horizontalparallaxe für die Sonne < 9”, für den Mond etwa 57’ im Mittel, in Urban and Seidelmann (2012) wird auf den Wert 1° für den Mond hingewiesen. Da der Unterschied nicht nur in der Höhe über dem Mittelpunkt (oder Schwerpunkt) liegt, sondern auch in der variierenden Geschwindigkeit wegen der Erdrotation, müssen diese Parameter bei genauer Bestimmung

berücksichtigt werden. Zur oben erwähnten Parallaxe käme auch noch eine Korrektur für die Abberation (diurnal aberration) hinzu. Eine Herleitung dieser Rechnung wird beispielsweise in Urban and Seidelmann (2012) demonstriert.

4.4.6. Zusammenfassung der Korrekturparameter

Für die vorgenannten Aspekte kann man folgende Richtwerte zusammenfassend ansetzen: Refraktion im Bereich zwischen 1' am Horizont und gegen 0 am Zenit. Die Präzession bewegt sich im Bereich von 50" pro Jahr, die Nutation ist deutlich geringer und beläuft sich auf weniger als 20" in ihrer Periode. Der Effekt der Abberation hat eine ähnliche Größenordnung wie die Nutation, für die Parallaxe kann man weniger als 1" ansetzen. Hinzu kommt die variable Eigenbewegung der Sterne, in aller Regel < 1" jährlich.

4.5. Rotation der Erde

Die Erdrotation ist für die Darstellung von Bedeutung, da die Winkelgeschwindigkeit ω insbesondere in der Nähe des Äquators eine Korrektur der Position der Sterne in regelmäßigen Abständen erfordert. Eine siderische Rotationsperiode der Erde beträgt näherungsweise 23 Stunden, 56 Minuten und 4 Sekunden. Würde man nun mit der Hololens am Äquator stehen und ein Sternbild beobachten, ergäbe sich eine Bewegung auf der Erde von grob 1670km/h, d.h. ein Objekt am Himmel zieht mit dieser Geschwindigkeit vorbei. Das entspricht ca 464m/s. Wie das Bild 11 zeigt, bewegen sich die Sterne - in Abhängigkeit vom Standpunkt auf der Erde - in bestimmten (kreisförmigen) Bahnen und ziehen Spuren. Wer sich mit Astrophotographie beschäftigt, weiß, dass längere Belichtungszeiten die Sterne verschmieren und eben diese Rotation der Erde als Spur auf dem Bildern zurücklassen, daher ist bei längeren Belichtungszeiten eine Nachführung nötig. Eine Faustregel, die zur Orientierung dient, liefert eine Nachführungszeit von 5 Sekunden bei einer Brennweite um die 80mm Kleinbildformat. Um die scheinbare Bewegung der Sterne am Himmel möglichst korrekt in der Hololens zu korrigieren, ist also eine Nachberechnung der Position nötig.

Für diese Anwendung ist die Betrachtung der Erde als Kugel ausreichend genau. Der Radius R der Erdkugel beträgt etwa 6370 km. Der Umfang ergibt sich daher mit $U = 2R\pi$ und beträgt in dieser Rechnung gerundet 40 024 km. Mit $v = s/t$ ergibt sich am Äquator für näherungsweise 24 Stunden eine Geschwindigkeit von 1667,7km/h, also die oben erwähnten 464m/s. Mit dem Cosinus von ϕ können für einen beliebigen Breitenkreis daher die Geschwindigkeiten bestimmt werden. Am Beispiel Wien mit $\phi = 48.2$ Grad nördliche Breite ergibt sich ein Radius von

$$\phi_r = \frac{R}{\cos(48.2)} = 4246km \tag{22}$$

der Umfang ist daher 26678km und mit der Formel für v ergibt sich eine Geschwindigkeit von gerundet 309m/s. Stünde man in Norwegen auf einer Höhe von etwa 70° nördlicher Breite, beträgt v nur noch etwa 215m/s. Für den Nordpol im Sinne der Rotationsachse ergäbe sich daher ein Wert von 0, da $\cos(90) = 0$ und man erhält für den Radius $\phi_r = 0$ entsprechend den selben Wert. Für die Anwendung soll der Breitengrad ϕ dahingehend berücksichtigt werden, dass die Aktualisierungsfrequenz dem Breitengrad des Anwenders angepasst wird, wobei in unserem

BSC ID	RA (°)	DEC (°)
1035	52,26	59,94
1040	52,47	58,87
1148	57,58	71,33
1155	57,38	65,52
1542	73,51	66,34
1568	74,32	53,75
1603	75,85	60,44

Tabelle 6: Sterne des Sternbilds Giraffe mit Angaben für die Koordinaten bzw. der Nummer (ID) aus dem BSC-Katalog, Koordinaten in Grad.

Fall eine einfache lineare Tabelle als Näherung ausreichend ist. Angemerkt sei weiters, dass die Rotationsgeschwindigkeit natürlichen Schwankungen unterliegt, aber diese Werte sind rein akademischer Natur und bewegen sich im Millisekunden-Bereich[4].

4.6. Projektion in die Ebene

Entsprechend den kartografischen Erkenntnissen aus den vergangenen Jahrhunderten lässt sich eine Fläche auf der Kugel nicht ohne weiteres auf die Ebene übertragen. Daher wird für Darstellungen auf Karten eine Projektion verwendet, die aber nie flächen-, längen- und winkeltreu zugleich sein kann. Abstriche müssen immer gemacht werden, daher ist keine Karte fehlerfrei. Koordinaten eines Sterns sind in aller Regel in sphärischen Werten angegeben. Zur Darstellung auf einem Blatt Papier benötigt es aber die Umrechnung in eine ebene Fläche. Für die Astronomie haben sich zwei Typen von Projektionen etabliert: für großflächige Darstellungen, die den gesamten Sternenhimmel abdecken soll, verwendet man die Mollweide-Projektion, für kleinräumige Illustrationen bzw. für nur eine Hälfte der Himmelskugel, beispielsweise die nördliche Hemisphäre, eine stereographische Projektion. Entsprechend dem Anforderungsprofil 5.1 ist nur eine Projektion für kleinräumige Strukturen zu überdenken. Die Frage ist daher, ob für den Genauigkeitsanspruch für diese Arbeit überhaupt eine kartographische Projektion notwendig ist, welche Eigenschaften sie gegebenenfalls haben soll oder ob die Umrechnung von Kugelkoordinaten dem Umsetzungsgedanken genügt. Wir illustrieren das Problem am Sternbild Giraffe. Gemäß der Tabelle von Sluys enthält das Sternbild die Sterne nach Tabelle 6. Wir rechnen diese aus den äquatorialen Koordinaten um, d.h. wir gehen von sphärischen Koordinaten auf Koordinaten der stereographischen Projektion sowie rechtwinkelige Koordinaten über, um die Ergebnisse vergleichen zu können.

Hierfür verwenden wir folgenden Formelsatz für die kartesischen Koordinaten, der bereits im Abschnitt Koordinatensysteme (siehe 4.2) erwähnt wurde:

$$\begin{aligned} x &= r\cos\delta\cos\alpha \\ y &= r\cos\delta\sin\alpha \\ z &= r\sin\delta \end{aligned} \tag{23}$$

[4]https://www.tuwien.at/tu-wien/aktuelles/news/news/die-erde-dreht-sich-schneller

RA (°)	DEC (°)	x stereogr.	y stereogr.	x kart.	y kart.	z kart.
52.47	58.87	0.79	1.20	0.52	0.678	0.51
52.26	59.94	0.81	1.21	0.52	0.68	0.50
57.38	65.52	0.80	1.37	0.49	0.76	0.41
57.58	71.33	0.86	1.44	0.50	0.79	0.32
73.51	66.34	0.46	1.72	0.25	0.87	0.40
75.85	60.44	0.37	1.73	0.21	0.84	0.49
74.32	53.75	0.37	1.66	0.21	0.77	0.59

Tabelle 7: Die Tabelle listet die Koordinaten im IRCS System (Epoche J2000.0) auf sowie die ermittelten Koordinaten für die stereographische Projektion und die Koordinaten für die Umrechnung in ein kartesisches Koordinatensystem. RA und DEC in Grad, stereographische und kartesische Koordinaten als x, y - Werte gerundet auf 2 Nachkommastellen.

Das r steht hier für den Radius bzw. die Distanz, wir nehmen aber 1 (Einheitskreis) an, da die Tiefe der Objekte für die Darstellung keine Rolle spielt. α ist wieder die Rektaszension, δ ist die Deklination. Die Ermittlung der Koordinaten für die stereographische Projektion ist gegeben durch folgende Gleichungen, die dem Buch Snyder (1987) entnommen sind:

$$x = k \cos\phi \sin(\lambda - \lambda_0) \tag{24}$$
$$y = k[\cos\phi_1 \sin\phi - \sin\phi_1 \cos\phi \cos(\lambda - \lambda_0)]$$

Die Konstante k ist gegeben durch

$$k = \frac{2R}{1 + \sin\phi_1 \sin\phi + \cos\phi_1 \cos\phi \cos(\lambda - \lambda_0)} \tag{25}$$

Die Variablen ϕ_1 und λ_0 sind die Werte für das Zentrum, k ist ein Skalierungsfaktor (hier $k = 1$), R als Radius, die Ausrichtung ist als Draufsicht (polar) ausgelegt. Die Berechnung erfolgt Nach Berechnung ergibt sich folgende Tabelle 7 mit den entsprechenden Werten.

Mit Hilfe von pyplot wird die kleine Tabelle geplottet (siehe Grafik 13), diese zeigt Unterschiede in der Darstellung. Der Versatz im Sinne der Lage ergibt sich entsprechend der Formeln; die Grafik zeigt, dass eine gewisse Ähnlichkeit der Figur erhalten bleibt, es aber sehr wohl erkennbare Unterschiede und Verschiebungen gibt. Daraus folgt, dass für eine Abbildung in die Ebene durchaus eine Projektion zu verwenden wäre, die nicht zwingend stereographisch sein muss. Möglich wären auch eine konforme Lambert-Projektion oder ein UTM-Netz (Universal Transverse Mercator Grid). Da sich im späteren Verlauf der Anwendungsentwicklung aber zeigt, dass die Sterne im Sinne der Verortung auf der Kugel nur verschoben werden und nicht für den Blick durch die HL2 auf eine ebene Fläche übertragen werden müssen, kommt in der späteren Anwendung keine Projektion zum Einsatz.

Unabhängig der mathematischen Fragestellung ist zu überlegen, welchen Maßstab die Darstellung haben muss, um in der Hololens noch mit brauchbarer Erkennbarkeit der Sterne angezeigt zu werden. Als Analogie dient der Maßstab in topographischen Karten: eine Übersichtskarte,

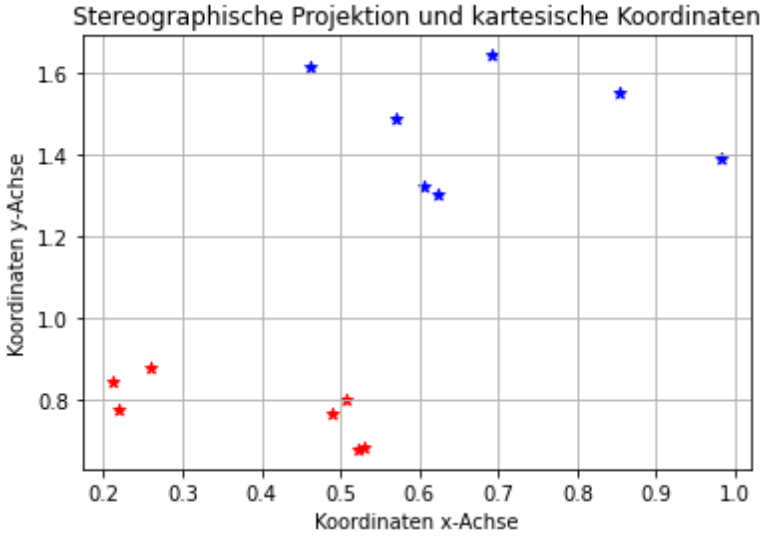

Abbildung 13: Die Grafik zeigt die errechneten Koordinaten für die stereographische Projektion, diese sind durch die blauen Sterne dargestellt. Die roten Sterne zeigt die Verortung für die Koordinatenwerte nach Übergang von sphärischen auf kartesische Koordinaten.

beispielsweise im Maßstab 1:1 Million, gewährt einen Überblick mit notwendigen wichtigen Signaturen, aber keine Details. Eine Karte im Maßstab 1:25 000 zeigt viele topographische Details, kann aber nur ein Ausschnitt der gesamten Region sein. Gleiches gilt für die Darstellung in der App: bei einem kleinen Maßstab (Übersicht) lässt sich das einzelne Sternbild nicht mehr erkennen und die Projektion muss für die Fläche gedacht werden (Mollweide -Projektion). Bei einem großen Maßstab ist nur der Ausschnitt zu sehen, den das FoV der Hololens erlaubt, womit man zur Frage gelangt, was überhaupt der optimale Maßstab ist. Da Sterne am Nachthimmel sehr klein sind, die gleiche Größe als Sterne in der Hololens aber kaum sichtbar ist, ergibt sich zwangsweise eine Vergrößerung der Objekte und dadurch ein tendenziell ungewohntes Bild, etwas überzeichnet mit Bild 31 auf Seite 64 dargestellt. Diese Darstellung ist stark überzogen und die Größe fließt in dieser Form nicht in die Anwendung ein.

4.7. Berechnungsschritte

Zur Berechnung der Position eines Sterns kann der komplexe Algorithmus wie in 4.4 dargelegt, verwendet werden. Eine einfache Variante für die grobe Positionierung, basierend auf nachfolgenden Berechnungsschritten, ist für die Hololens-Anwendung aber ausreichend.

Die Berechnung beinhaltet:

1. Ermitteln der UTC
2. Berechnen des Julianischen Datums 0h UTC
3. Sternzeit für Greenwich für 0h UTC und t

4. Sternzeit für den Beobachtungsort
5. Stundenwinkel des Sterns
6. Umrechnung in das gewünschte Koordinatensystem

4.7.1. UTC

Die UTC wird mittels der in die Entwicklungsumgebungen integrierten Klassen ermittelt, um die für den Moment korrekte Zeit zu erhalten. Für die manuelle Berechnung zum Testen beliebiger Sterne zwecks Qualitätskontrolle wird ein beliebiger Zeitpunkt gewählt.

4.7.2. Julianisches Datum

Das Julianische Datum (oder eigentlich Julianischer Tag, die Begriffe werden in der Literatur nicht einheitlich verwendet) ist eine Definition für die Zählweise von Tagen seit einem bestimmten Zeitpunkt im gregorianischen Kalender. Definitionsgemäß beginnt die Zählung im Jahre -4712 zur Mittagszeit, also entsprechend 12:00 im Sinne der UTC. Der Vorteil dieser Definition liegt darin, dass man mit einem fixierten Datum einfache Zählungen für Berechnungen durchführen kann.

Die Berechnung des JD erfolgt gemäß dem Algorithmus aus dem Werk von Meeus, siehe Meeus, 1998: Für diese Ausführung gilt: J = Jahr, M = Monat, D = Tag. Da für diese Arbeit nur aktuelle und zukünftige Tage in Betracht kommen und keine Rückrechnung für beliebige Zeiten vorgesehen ist, kann der Abschnitt mit den Tageskorrekturen vernachlässigt werden, wird aber der Form halber angefügt.

Für die Berechnung vor der Implementierung wird Python verwendet, das Skript basiert auf einer Ausgabe von ChatGPT 3.5 (Stand November 2023):

Listing 1: Python - Julianisches Datum

```
def julian_date(year, month, day, utc=0):
global jd
global t

if month > 2:
y = year
m = month
else:
y = year - 1
m = month + 12
d = day
h = utc/24
if year <= 1582 and month <= 10 and day <= 4:
b = 0
elif year == 1582 and month == 10 and day > 4 and day < 15:
# Gregorian calendar reform: 10 days (5 to 14 October 1582) were skipped.
# In 1582 after 4 October follows the 15 October.
d = 15
b = -10
else:
# Gregorian Calendar
a = int(y/100)
b = 2 - a + int(a/4)
```

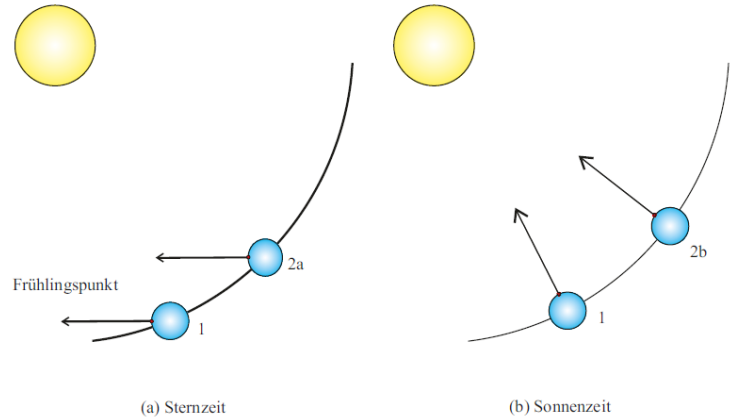

Abbildung 14: Die Grafik illustriert die Bewegung der Erde um die Sonne und die daraus resultierende Differenz der Tageslänge. Quelle: wikipedia

```
24    jd = int(365.25*(y+4716)) + int(30.6001*(m+1)) + d + h + b - 1524.5
25    return(jd)
```

Der vorangegangene Codeschnipsel hat als Rückgabewert das Julianische Datum, Beispiel für 27. Mai 2023: 2460091.5. Für den selben Tag um 10:09:00 UTC liefert das Skript als Wert 2460091.920. Ein Vergleich mit den Tool Ephemeris Computation Office, NAOJ (ECO) liefert das selbe Ergebnis mit einer kleinen Abweichung für den Wert um 10:09:00 UTC, der sich aus der Differenz zwischen UTC und UT ergibt.

4.7.3. Sternzeit

Um die aktuelle Position eines Sterns am Himmel ermitteln zu können, ist seine Sternzeit notwendig. Die Sternzeit ist eine fortlaufende Zeitangabe für jeden Tag, definiert über die Dauer, die die Erde für eine Rotation benötigt, um wieder am Frühlingspunkt durchzulaufen; sie wird also als Stundenwinkel des Frühlingspunkts definiert.

Im Abschnitt 4.3.1 wurde ein Überblick über die Definition der Zeit gegeben, nun betrachten wir die astronomischen Aspekte genauer. Grundsätzlich unterscheidet man zwischen siderischem und synodischen Tag. Im Falle des siderischen Tages betrachtet man die Rotationsdauer bezogen auf den Frühlingspunkt, diese Rotation dauert in etwa 23 Stunden, 56 Minuten und 4 Sekunden. Wie im Abschnitt 4.4 gezeigt, schwankt zwar die Rotation im Bereich der Millisekunden, aber diese Ungenauigkeit wollen wir an dieser Stelle ignorieren. Bei scheinbaren Lauf der Sonne um die Erde zeigt sich, dass diese Umdrehung etwas länger dauert, nämlich 24h. Die Begründung liegt darin, dass bei der tatsächlichen Bewegung der Erde um die Sonne das zurückgelegte Bogenstück berücksichtigt werden muss. Das Bild mit der Nummer 14 veranschaulicht diesen Sachverhalt.

Um also einen siderischen Tag mit einem synodischen Tag (und der entsprechenden Zeitrechnung) verknüpfen zu können, muss die Differenz bekannt sein. Für die Berechnung der Sternzeit, also jenen Zeitpunkt am Standort des Beobachters, zu welchem ein Stern wo sichtbar ist,

benötigt man die Sternzeit für Greenwich um 0 Uhr UTC, die Sternzeit für die Beobachtungszeit nach Greenwich und die Sternzeit für den Beobachtungsort.

Sternzeit Greenwich 0 Uhr Für die Sternzeit in Greenwich um 0 Uhr wurde von der IAU folgende Formel definiert, die in Sekunden, Stunden oder in Grad angeschrieben werden kann:

$$\Theta = 100,46061837 + 36000,770053608 * t + 0,000387933 * t^2 - \frac{t^3}{38710000} \tag{26}$$

Die obige Formel ist in Grad ° angegeben. Für die Variante in h kann man schreiben:

$$\Theta = (6 + \frac{41}{60} + \frac{50,5484}{3600}) + (\frac{8640184,812866}{3600} * t^2) + (\frac{0,09310}{3600} * t^2) - (\frac{0,0000062}{3600} * t^3) \tag{27}$$

Ergänzend noch die Formel in der Sekundendarstellung:

$$\Theta = 24410,54841 + 8640184,812866 * t + 0,0093104 * t^2 - 0,0000062 * t^3 \tag{28}$$

Aus den obigen Formeln ergibt sich in Vielfaches von 360°, daher muss der Anteil um die Umrundungen reduziert und entsprechend mit dem Rest ($\theta < 1$) gerechnet werden. Die vorgenannten Formeln sind mittlerweile durch eine andere Formulierung, basierend auf einer Neudefinition der UT1, adaptiert worden. Für die Genauigkeit, die gefordert ist, genügt aber obige Darstellung, da die Neuformulierung im Sinne der Genauigkeit für diese Arbeit nicht relevant ist. Die Neudefinition beinhaltet zwei Argumente der Zeit (TT und UT1) und kann in Urban and Seidelmann (2012) nachgelesen werden.

Sternzeit Greenwich für die Beobachtungszeit Um die Sternzeit am Beobachtungszeitpunkt zu ermitteln, wir Θ um die Zeit nach UT ergänzt und mit einem Faktor $c = 1,00273790935$ multipliziert. Der nur hier genannte Faktor c ergibt sich aus dem Unterschied zwischen synodischen und siderischer Tageslänge. Beispiel: der siderische Tag mit seiner definierten Länge von 23h 56min und 4s wird mit dem Faktor c multipliziert, d.h. 86164 Sekunden * 1.00273790935 = 86400 Sekunden, was der Länge eines synodischen Tages entspricht.

Für einen fiktiven Beobachtungstag um 03:00 UTC gilt somit für Θ_t:

$$\Theta_t = \Theta + 3h * c \tag{29}$$

Sternzeit am Beobachtungsort Um die Sternzeit am Beobachtungsort zu erhalten, muss für Θ_t noch der jeweilige Längengrad des Beobachters ergänzt werden. Wir nehmen Wien als Beispiel und erhalten für die geographische Länge λ einen Wert von 16.36° östliche Länge. Die Sternzeit am Beobachtungsort ergibt sich somit durch

$$\Theta_{local} = \Theta_t + / - \lambda \tag{30}$$

wobei für λ gilt, dass $\lambda < 0$ für westlich von Greenwich und $\lambda > 0$ für östlich von Greenwich.

Zur Überprüfung der errechneten Werte für das Julianische Datum, die GMST und LMST wird der Rechner der ECO herangezogen. Die Werte müssen für unterschiedliche Datumsangaben und Uhrzeiten korrekt sein, da der Stundenwinkel sonst nicht richtig errechnet wird, woraus sich Fehler in den Koordinaten ergeben würden. Beispielhaft sei der 1. November 2023 für 0:00 UTC mit den Uhrzeiten von +0 LST, + 3 LST und -5 LST verglichen. Für den ersten Zeitpunkt liefert die obige Berechnungsformel einen Wert von $GMST = 40,028$ und deckt sich mit dem Ergebnissen der Datenbank der (NOAJ, 2023). Für den selben Tag für 3:00 Uhr LMST erhalten wir $LMST = 354.904$ und für die lokale Zeit -05:00 (also westwärts) erhalten wir $LMST = 115.233$. Für Werte abseits 0:00 UTC verschieben sich die Ergebnisse entsprechend der Drehung um 15° / Stunde. In aller Regel sind die Werte bis zur 4. Kommastelle korrekt, danach weichen sie ab. D.h. man kann bei der Berechnung der GMST bzw. LMST von einer hohen Genauigkeit ausgehen.

4.7.4. Stundenwinkel

Im ortsfesten, äquatorialen Koordinatensystem definiert der Stundenwinkel in der Äquatorebene eine Koordinate eines Sterns. Zusammen mit der Deklination, die im festen wie auch im rotierenden äquatorialen Koordinatensystem gleich ist, bestimmt der Stundenwinkel die Koordinaten des Sterns. Der Zusammenhang zwischen rotierend und ortsfest ergibt sich aus

$$H_\alpha = \theta - \alpha = \theta_0 - \lambda - \alpha \tag{31}$$

4.7.5. Koordinatentransformation

Die für die Sterne verwendeten Koordinaten sind äquatoriale Koordinaten, die vom rotierenden ins ruhende System umgerechnet werden. Um diese von einem Standpunkt auf der Erdoberfläche verwenden zu können, müssen sie vom äquatorialen System ins Horizontsystem umgerechnet werden. Die Ansätze dafür sind eine Rotationsmatrix oder die Benutzung der sphärischen Trigonometrie, konkreter die Berechnung über ein sphärisches Dreieck (auch nautisches oder astronomisches Dreieck). Gegeben sind also die Rektaszension und die Deklination eines Sterns zu einer bestimmten Epoche, über die vorhin durchgeführten Berechnungen haben wir weiters den Stundenwinkel t ermittelt und wir benötigen die geographischen Koordinaten ϕ und λ des Beobachters. Gesucht sind Azimut A und Höhe h im Zielkoordinatensystem, also dem horizontalen Koordinatensystem. In den meisten Fällen wird die Berechnung mit Ausrichtung Süden verwendet, in einigen Teilbereichen der Astronomie erfolgt eine Ausrichtung beginnend bei Norden, siehe Hanslmeier (2020). Für diese Arbeit wird eine Nordorientierung gewählt. Die Berechnung folgt der Herleitung aus Montenbruck (2005) und Richter (2005).

Das nautische Dreieck auf der Kugel ist definiert über die Punkte des Pols, Zenit und den Sternort, was einer Definition im Sinne eines allgemeinen sphärischen Dreiecks entspricht. Die Abbildung 15 zeigt die wichtigsten Winkel und Parameter im astronomischen Kontext. Für die Berechnung benötigt man den sphärischen Seitenkosinussatz sowie den Sinussatz. Ersterer ist allgemein für ein beliebiges (eulersches) Dreieck auf der Kugel definiert durch

$$\cos a = \cos b \cos c + \sin b \sin c \cos \alpha \tag{32}$$

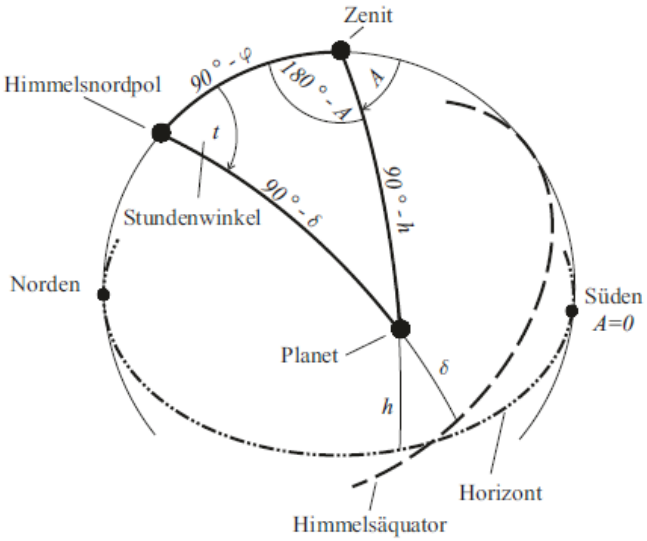

Abbildung 15: Die Darstellung zeigt das nautische Dreieck mit den entsprechenden Koordinatenpunkten (Zenit, Pol, Sternort bzw. in dieser Grafik Planet) und die relevanten Winkel und die Horizont- bzw. Äquatorlinie. Grafik: Dieter Richter, Ephemeridenrechnung Schritt für Schritt

.

Der Seitenkonsinussatz kann zyklisch vertauscht werden, daraus ergibt sich zusätzlich $\cos b = \cos\alpha\cos c + \sin c \sin\alpha\cos\beta$ bzw. $\cos c = \cos\alpha\cos b + \sin\alpha\sin b\cos\gamma$.

Der Sinussatz ergibt sich als

$$\frac{\sin a}{\sin\alpha} = \frac{\sin b}{\sin\beta} = \frac{\sin c}{\sin\gamma} \tag{33}$$

Die Gleichungen 32 und 33 gelten für das sphärische Dreieck in allgemeiner Notation.

Für die Höhe h über dem Horizont ergibt sich über das nautische Dreieck folgende Formel:

$$\sin h = \sin\phi\sin\delta + \cos\phi\cos\delta * \cos t \tag{34}$$

Das Ergebnis muss zwischen $-90°$*und* $+90°$ liegen, da wir hier die Höhe über dem Horizont berechnen. Das Ergebnis ist mathematisch eindeutig und bedarf in diesem Fall keiner Fallunterscheidungen.

Für den Azimut ergeben sich folgende zwei Gleichungen:

$$\sin A = \frac{\cos\delta\sin t}{\cos h} \tag{35}$$

bzw

$$\cos A = \frac{-\sin\delta\cos\phi + \cos\delta\cos t\sin\phi}{\cos\alpha} \tag{36}$$

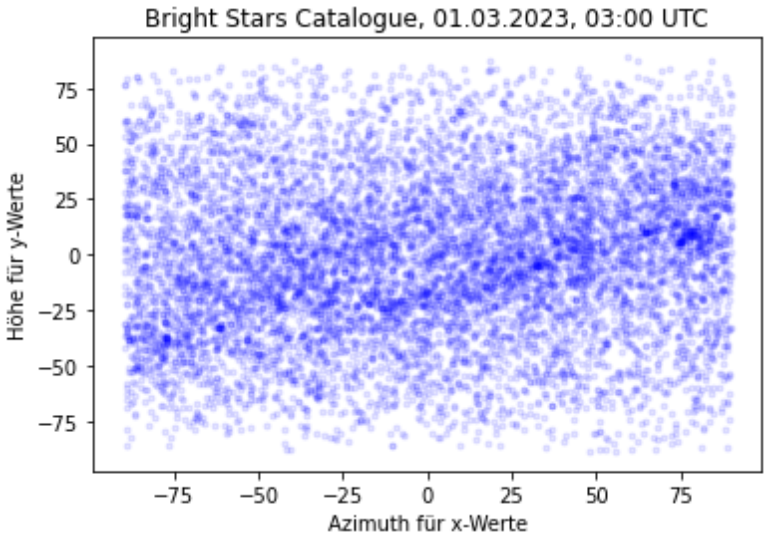

Abbildung 16: Die Darstellung zeigt die Sterne bei ungenauer Berechnung des Azimut für alle Sterne aus dem BSC-Katalog. Der Azimut liegt im Bereich zwischen -90 und 90°.

Diese Gleichung ist mathematisch aber nicht eindeutig, sodass es einer Fallunterscheidung bzw. einer zweiten Gleichung bedarf. Die Umkehrfunktion des Sinus ist zwar im Interval $[-1, 1]$ definiert, der Wertebereich beschränkt sich aber auf das Intervall $-\pi/2, \pi/2$, sodass man prüfen muss, in welchem Bereich das Ergebnis der Berechnung endet. Beim Durchrechnen der Werte ohne diese Fehlerprüfung verteilen sich die Sterne nur in einem Intervall zwischen -90 und 90 Grad Azimut, siehe Darstellung 16. Wie man der Grafik entnehmen kann, ist die Verteilung nicht über den Vollkreis gegeben, sondern nur in einem Wertebereich zwischen -90 und 90, entsprechend der Werte für die Umkehrfunktion arcsin(). Bei einer händischen Berechnung wäre entsprechend den Ergebnissen der Quadrant für die Resulate zu prüfen und dadurch die Postion zu berechnen. Bei einer Berechnung mit dem Computer nutzt man die $\arctan 2$ - Funktion, die eine Fallunterscheidung betreibt (siehe Hinweise in der Dokumentation zu numpy[5]) für Python. Damit wird die Quadrantenzuordnung automatisiert und man ergänzt bei negativen Werten den Vollkreis. Die Gleichung

$$\cos A = \frac{-\sin\delta\cos\phi + \cos\delta\cos t\sin\phi}{\cos\alpha} \tag{37}$$

entspricht x, die Gleichung

$$\sin A = \frac{\cos\delta\sin t}{\cos h} \tag{38}$$

entspricht y. Dadurch wird erreicht, dass die Wahl des Quadranten automatisiert erfolgt und die Berechnung korrekte Winkel liefert.

[5]https://numpy.org/doc/stable/reference/generated/numpy.arctan2.html

Nr	Rahmenbedingungen	Aufgaben / Eckpunkte
1	Ziel der Arbeit	Darstellung von Sternbildern HL2
2	Software / Hardware	Unity 2021 LTS Visual Studio 2022 Hololens Emulator Anaconda (Python)
3	Grundsätzliche Funktion der Anwendung	Eigene Anwendung in der Hololens Ermitteln der aktuellen Zeit/Datum Berechnung der Positionen Darstellung der Sterne
4	Qualitätsfragen	Lagegenauigkeit der Sterne

Tabelle 8: Auflistung der Mindestanforderung an den Demonstrator

5. Datenquellen und Positionsberechnung

Die Beobachtung der Sterne und das Auffinden der Sternbilder ist eine beliebte Beschäftigung im Dunstkreis astronomisch motivierter Hobbies, die durch viele Bücher und Beobachtungshilfen erleichtert wird. Es gibt Sternkarten, also drehbare Scheiben, die je nach Jahreszeit die Sterne am Himmel anzeigen beziehungsweise viele Bücher, die für jedes Jahr die Konstellationen und besondere Ereignisse erklären. Das ganze wird ergänzt um eine Vielzahl an Softwaretools und Programmen aller Art, die auf diversen Plattformen lauffähig sind. Genannt seien hier Apps für iOS oder Android, browserbasierende Anwendungen für den PC oder klassische Software für Hobbyzwecke oder natürlich professionelle Astronomieanwendungen, deren Anspruch an die Informationsqualität und Genauigkeit der Sternörter besonders hoch ist (siehe 4.4). Alle diese Informationsträger haben eines gleich: die Information, wann wo welche Sterne am Himmel zu sehen sind, muss enthalten sein. Wie man theoretisch zu dieser Kenntnis gelangt, wurde im vorangegangenen Abschnitt erläutert.

Nachfolgend wird mit der Programmiersprache Python der Algorithmus zur Berechnung eines Sternorts umgesetzt und die erhaltenen Daten werden mit Katalogwerten verglichen. Nach diesem Abschnitt wird im Kapitel Anwendungsprogrammierung die Umsetzung mit Unity und CSharp erläutert. Das nachfolgende Anforderungsprofil legt die Funktionen, die erfüllt sein sollen, dar.

5.1. Anforderungsprofil

Der für diese Arbeit erstellte Software-Prototyp soll folgende Mindestanforderungen gemäß Tabelle 8 erfüllen: die Anwendung muss beim Start die aktuelle Zeit bzw. das Datum ermitteln und die Sterne entsprechend dieser Parameter verorten, d.h. die Sternbilder sollen in der Anwendung dort zu sehen sein, wo sie mit freiem Auge auch erkennbar sind. Der Bezug ist entsprechend dem Koordinatensystem die Nordrichtung.

Die Basisfunktionalität setzt keine Verwendung einer GNSS-Einheit oder einer anderen Orientierungshilfe voraus. Im Zuge dieser Arbeit wird versucht, die IMU der Hololens auszulesen

und für eine Weiterentwicklung ist die Anbindung an einen Positionierungsdienst (D-GPS oder Smartphone) geplant.

5.2. Daten und Berechnungsgrundlagen

Für diese Arbeit werden frei verfügbare Daten verwendet. Dies betrifft den Sternkatalog, die Grenzen, die Sternbilder an sich (also jene Sterne, die die Sternbilder darstellen) sowie eine Liste mit den Abkürzungen der Sternbilder zur besseren Orientierung. Spezielle Grafiken kommen nicht zum Einsatz, die Sterne werden in der Software als herkömmliche 3D-Objekte generiert.

5.2.1. Sternkataloge

Die Astronomie kennt eine Vielzahl allgemeiner und spezifischer Kataloge, die für den jeweiligen Verwendungszweck optimiert sind. Wichtige Beispiele sind die Kataloge der Fundamentalsterne (FK4, FK5 oder FK6), der Hipparchos-Katalog oder die Datenkataloge der Gaia - Mission. Üblicherweise enthalten Sternkataloge die Positionen der Sterne bezogen auf eine bestimmte Epoche (zum Beispiel J2000.0), Informationen über die Helligkeit, die Spektralklasse, Entfernung, Radialgeschwindigkeit oder auch die Eigenbewegung eines Sterns, wenn diese bekannt ist. Die in Sternkatalogen angegebenen Koordinaten beziehen sich immer auf eine bestimmte Epoche. Hintergrund ist die genaue Kenntnis der Lage der Sterne, soweit dies möglich ist, in Verbindung mit den Parametern der Erde und deren Position im Raum. Des weiteren ist wichtig zu wissen, welche Koordinaten in den Katalogen verwendet werden: scheinbare, wahre oder mittlere Werte? In der Astronomie wird der Begriff scheinbar für die Position verwendet, die man quasi mit dem Auge wahrnimmt. Scheinbar deswegen, weil die optische Position nicht zwangsläufig koordinativ korrekt sein muss (siehe Abschnitt 4.4). Im Regelfall werden in den Katalogen mittlere Koordinaten angegeben. Naturgemäß existiert eine Vielzahl an Sternkatalogen, die für diese Arbeit in Betracht kommen. Die Wahl viel auf den Bright Stars Catalogue (BSC), da dieser mit ca 9100 Sternen, wie der Name vermuten lässt, mit einem hohen Wert für die Helligkeit (Magnitude) einhergehen. Der Mindestwert für die Helligkeit beträgt 6.5 mag, es sind also nur jene Sterne enthalten, die bei guten Verhältnissen auch mit dem freien Auge sichtbar sind. Diese Überlegung ist der Idee geschuldet, die dargestellten Sterne in der Hololens ggf. bei guten Sichtverhältnissen mit den tatsächlichen Sternpositionen vergleichen zu können. Für diese Arbeit wird die 5. Überarbeitung (5th Revised Ed.) verwendet (Hoffleit and Jaschek, 1991). Als Datenquelle dient das in der Astronomie gebräuchliche Portal „Centre de Données astronomiques de Strasbourg (Strasbourg astronomical Data Center)“. Der BSC wurde über den Suchdienst Vizier[6] geladen.

Die Eigenbeschreibung des Katalogs, Zitat:

> „The Bright Star Catalogue (BSC) is widely used as a source of basic astronomical and astrophysical data for stars brighter than magnitude 6.5. The catalog contains the identifications of included stars in several other widely-used catalogs, double- and multiple-star identifications, indication of variability and variable-star

[6] https://vizier.cds.unistra.fr/viz-bin/VizieR

identifiers, equatorial positions for B1900.0 and J2000.0, galactic coordinates, UBVRI photoelectric photometric data when they exist, spectral types on the Morgan-Keenan (MK) classification system, proper motions (J2000.0), parallax, radial- and rotational-velocity data, and multiple-star information (number of components, separation, and magnitude differences) for known nonsingle stars. In addition to the data file, there is an extensive remarks file that gives more detailed information on individual entries. This information includes star names, colors, spectra, variability details, binary characteristics, radial and rotational velocities for companion stars, duplicity information, dynamical parallaxes, stellar dimensions (radii and diameters), polarization, and membership in stellar groups and clusters. The existence of remarks is flagged in the main data file.“

Der BSC enthält viele Informationen bzw. Spalten, die für diese Arbeit nicht relevant sind. Wir extrahieren Rektaszension und Deklination aus der Epoche 2000.0. Damit haben wir eine Liste von 9100 Sternen mit entsprechenden Koordinatenwerten, die wir weiterverwenden können. Eine vollständige Auflistung der Inhalte findet sich im Anhang.

Ergänzend zum Sternkatalog selbst wurde ein weiterer Katalog importiert, der die Grenzen der Sternbilder gemäß Definition der IAU beschreibt. Dieser Katalog namens **Constellation Boundary Data : VI/49** enthält alle 88 Sternbildgeometrien in entsprechenden Epochen und kann analog zum BSC - Katalog über den Suchdienst Vizier geladen werden. Für diese Arbeit wurde wieder die Epoche J2000 verwendet, definiert wurden die Areale für die Epoche B1875. Die Darstellung in Bild 17 zeigt eine Veranschaulichung der Sternbildflächen aus der Aladin-Software[7]. Wie man auf diesem Bild auch erkennen kann, bedecken die Areale für die 88 Sternbilder die gesamte Kugeloberfläche. Die Sternbildgrenzen wurden also dahingehend definiert, dass keine freie Fläche, die unbestimmt ist, übrig bleibt.

Die Tabelle 9 zeigt einen Auszug aus der Koordinatenliste für die Sternbildbereiche. Die ersten beiden Spalten kennzeichnen die Rektaszension bzw. die Deklination für den jeweiligen Koordinatenpunkt, die dritte Spalte die Kurzbezeichnung des Sternbildes (hier And für Andromeda) und die letzte Spalte gibt an, ob es sich um einen Punkt aus der Originaldatei handelt oder der Koordinatenpunkt interpoliert wurde.

Was nun noch fehlt sind die Asterismen der Sternbilder, wie man sie erwarten würde, also eine Polylinie aus untereinander der jeweiligen Figur entsprechenden Linien. Hierfür wurde eine Zuordnungstabelle verwendet, die von Marc van der Sluys[8] veröffentlicht wurde. Die Tabelle ist unter einer CC by 4.0 Lizenz erhältlich. Die Tabelle enthält die Abkürzung des jeweiligen Sternbildes, die Anzahl der Sterne, die verwendet werden und die Nummern gemäß dem BSC. Somit ist eine Zuordnung gegeben und die entsprechenden Sternbilder können gezeichnet werden. Die erwähnten drei Datenkataloge (BSC, Sternbildgrenzen sowie die Sternbildfiguren mit Referenz zum BSC-Katalog) fließen in die Anwendung als Grundlage ein. Ergänzend zu den drei Katalogen kommt noch eine Tabelle mit näherungsweise Mittelpunktskoordinaten dazu, um die Sternbildbereiche auch mit einer Überschrift (label) versehen zu können. Diese Tabelle enthält nur je Sternbild ein Koordinatenpaar und die entsprechende Sternbildbezeichnung. Diese Tabelle dient in erster Linie zu Kontrollzwecken.

[7] https://aladin.cds.unistra.fr/
[8] http://hemel.waarnemen.com

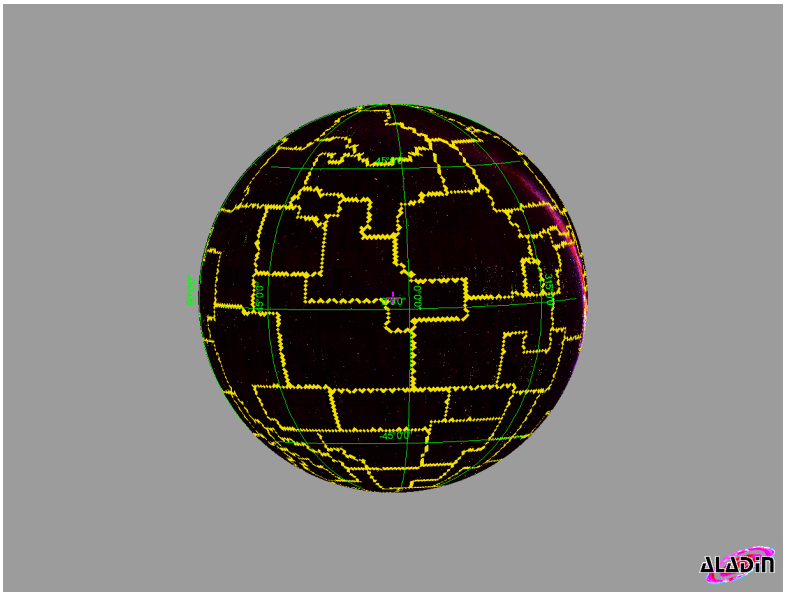

Abbildung 17: Darstellung der VI/49 Katalogdaten in der Software Aladin. Die Grafik zeigt die Abdeckung der jeweiligen Sternbildbereiche auf der Kugel. Wie man auf Grund der Linien erkennen kann, sind die Bereiche keine Polygone basierend auf wenigen Punkten, sondern bestehen aus vielen einzelnen Koordinatenpunkten um die gesamte Oberfläche abdecken zu können. Quelle: Aladin

RA	DEC	Constellation	Source
23.5357132	+35.1897736	AND	I
23.4684925	+35.1880264	AND	I
23.4012642	+35.1860695	AND	I
23.3340359	+35.1838989	AND	I
23.2668228	+35.1815186	AND	I
23.1996002	+35.1789322	AND	I
23.1323795	+35.1761322	AND	I
23.0651703	+35.1731300	AND	I
22.9979553	+35.1699181	AND	I
22.9643536	+35.1682358	AND	O

Tabelle 9: Auszug aus der Tabelle der Sternbildgrenzen des Katalogs VI/49. Rektaszension und Deklination sind in Dezimalgrad angegeben, die Abkürzung entspricht der offiziellen Tabelle gemäß der IAU, siehe 19.

Typ / Parameter	Werte gerundet
Uhrzeit	01:00 UTC
Julianisches Datum 0 Uhr	2460149,5
t	0.2355783710
Sternzeit Greenwich GMST	301,46
Sternzeit Greenwich GAST	301,46
Sternzeit Beobachtungszeit	316,46
Sternzeit am Beobachtungsort	300,13
Stundenwinkel	243,75
Höhe h	34,37
Azimut A	22,67

Tabelle 10: Berechnungsbeispiel für den Stern Alpha Cam aus dem Sternbild Giraffe für den 24. Juli 2023 um 01:00 UTC.

5.2.2. Berechnungsbeispiel

Mit den Erkenntnissen der vorigen Abschnitte wollen wir für den Stern Alpha Cam für den 24. Juli um 01:00 UTC am Standort Wien das julianische Datum, die Sternzeit sowie Azimut und Winkel berechnen. Der Stern hat für die Epoche J2000 die ICRS-Koordinaten RA 04 54 03.0141123 sowie Dec +66 20 33.619968. Diese müssen zuerst auf die aktuelle Zeit, also die geozentrischen Koordinaten für den gewählten Zeitpunkt, aktualisiert werden. Das julianische Datum für den 24. Juli um 0:00 Uhr ist 2460149.50000, t = 0.2355783710. Die GMST für diesen Zeitpunkt, auch wieder um 0:00 Uhr, ist 20h 05min 51.213s. Für die Uhrzeit 01:00 benötigen wir noch den Korrekturfaktor, d.h. 1 * 1.00273790395, also die GMST + 1.00273790395. Die Sternzeit ergibt sich für den Standort Wien als LMST = GMST + λ / 15. Mit dem Stundenwinkel H_α können dann die Umrechnungsformeln befüllt werden. Die Tabelle 10 zeigt das Ergebnis der Berechnung. Das Ergebnis dieser Rechnung wird mit den Werten aus Stellarium und dem ECO-Tools des NOAJ verglichen. Für eine Darstellung in der Hololens sind die ermittelten Werte für die Höhe über dem Horizont und den Azimutwinkel ausreichend genau.

Für die Umrechnung der Koordinaten aller Sterne wurde auf Python zurückgegriffen. D.h. der Test der Umrechnungsqualität (Genauigkeit) wird mit Python durchgeführt. Hierfür werden einzelne Stichproben gezogen bzw. Verlaufskurven und Plots zum Vergleich mit unterschiedlichen Datenquellen erstellt. Danach wird im Abschnitt 6 der Code in C# für Unity übersetzt. Da die Anwendung die aktuelle Zeit ermitteln soll, wird der Code so aufgebaut, dass einerseits die erforderlichen Parameter abgefragt werden, aber andererseits auch eine beliebige Zeit zwecks Genauigkeitsprüfungen, genutzt werden kann. Für die gesamte Liste muss auch berücksichtigt werden, dass alle Sterne im Süden eine negative Deklination aufweisen und die Werte für λ in Abhängigkeit der Lage zu Greenwich positiv oder negativ zu verwenden sind. Um die Berechnung zu überprüfen, werden mehrere Sterne gewählt, die entsprechenden Ergebnisse sind im Anhang aufgelistet.

Für die Berechnung der Positionen wurde nachfolgendes Skript verwendet:

Listing 2: Python - Skript zur Berechnung der Positionen

```
1
2  import time
3  import numpy as np
4  import matplotlib.pyplot as plt
5  import csv
6
7  # switch aktuelles oder bestimmtes Datum und Zeit mit UTC
8
9  holo_time= "";
10 Ort = "Wien"
11
12 # Nummern von Teststernen entsprechend BSC
13
14 # 1541 = alpha cam
15 # 7923 = deneb
16 # 5055 = Spica
17 # 1712 = Rigel
18 # 3981 = Regulus
19 teststern = 7923
20
21 # Aktuelle Zeit oder bestimmtes Datum
22 if(holo_time == "var"):
23 jahr = 2024
24 monat = 2
25 tag = 19
26 utc = 20
27 else:
28 stempel = time.localtime()
29 jahr = stempel.tm_year
30 monat = stempel.tm_mon
31 tag = stempel.tm_mday
32 utc = stempel.tm_hour
33 utc_minuten = stempel.tm_min
34 utc_sekunden = stempel.tm_sec
35
36 # Ort bestimmen
37 if (Ort == "Wien"):
38 phi = 48.2
39 lam = 16.36
40 elif (Ort == "Sydney"):
41 phi = -33.87
42 lam = 151.22
43 elif (Ort == "NY"):
44 phi = 40.72
45 lam = -74.02
46 elif (Ort == "Buenos"):
47 phi = -34.6
48 lam = -58.3
49 elif (Ort == "Toronto"):
50 phi = 43.5
51 lam = -79.2
52 elif (Ort == "Tokyo"):
53 phi = 35.65
54 lam = 139.74
55
56 # Liste des BSC Katalogs
57 data_sterne = np.genfromtxt('liste.csv', delimiter=';', dtype='f')
58 c_sterne1 = data_sterne[:,0]
59 c_sterne2 = data_sterne[:,1]
60
61 # julianisches Datum basierend auf Meeus - Algorithmus + chatgpt
```

```
def julianisches_Datum(jahr, monat, tag, utc=0):

global jd
global t

if monat > 2:
y = jahr
m = monat
else:
y = jahr - 1
m = monat + 12
d = tag
h = utc/24
if jahr <= 1582 and monat <= 10 and tag <= 4:
b = 0
elif jahr == 1582 and monat == 10 and tag > 4 and tag < 15:
d = 15
b = -10
else:
a = int(y/100)
b = 2 - a + int(a/4)
jd = int(365.25*(y+4716)) + int(30.6001*(m+1)) + d + h + b - 1524.5
return(jd)

def siderial_time(jahr, monat, tag, utc=0):
jd = julianisches_Datum(jahr, monat, tag)
t = (jd - 2451545.0)/36525
st = ((24110.54841 + 8640184.812866*t +
0.093104 * t**2 - 0.0000062 * t**3)/3600)*15
faktor = st/360
f = faktor % 1
st = f * 360
return(st)

jd = julianisches_Datum(jahr, monat, tag)
t = (jd - 2451545.0) / 36525
gmst = siderial_time(jahr, monat, tag, utc=0)

if(holo_time == "var"):
lmst = gmst + ((utc * 15) * 1.00273790935)
else:
lmst = gmst + (((utc * 15) + (15*utc_minuten)/60 ) * 1.00273790935)

if(lmst < 0):
lmst += 360
elif(lmst > 360):
lmst -= 360

#Koordinaten der Sterne zum Zeitpunkt xy
m = 3.075
n = 1.336
for delta_ra in data_sterne:
delta_ra = m + n*(np.sin(np.radians(c_sterne1)) * np.tan(np.radians(c_sterne2)))

for delta_dec in data_sterne:
delta_dec = n * (np.cos(np.radians(c_sterne1)))

for stundenwinkel in c_sterne1:
stundenwinkel = lmst - (c_sterne1 + delta_ra)
neg_stundenwinkel = stundenwinkel < 0
stundenwinkel[neg_stundenwinkel] += 360

```

```
# # Koordinatenumrechnung
def height (phi, Dec, stundenwinkel):

sin_h = np.sin(np.radians(phi)) * np.sin(np.radians(c_sterne2 + delta_dec)) + \
np.cos(np.radians(phi)) * np.cos(np.radians(c_sterne2 + delta_dec )) * \
np.cos(np.radians(stundenwinkel))
h = np.arcsin(sin_h)
h = np.degrees(h)
return(h)

def azimut_atan (phi, Dec, Stundenwinkel, h):

x = (-np.sin(np.radians(c_sterne2 + delta_dec)) * np.cos(np.radians(phi)) + np.cos(np.
    radians(c_sterne2 + delta_dec)) * \
np.cos(np.radians(stundenwinkel)) * np.sin(np.radians(phi))) / np.cos(np.radians(h))

y = (np.cos(np.radians(c_sterne2 + delta_dec)) * np.sin(np.radians(stundenwinkel))) /
    np.cos(np.radians(h))
werte = np.arctan2(y, x)
werte = np.degrees(werte)
return (werte)

def kart_Koordinaten(hoehe_h, atan_wert):
radius_schale = 5
x_cart = np.round((radius_schale * np.cos(hoehe_h) * np.cos(atan_wert)), 3)
y_cart = np.round((radius_schale * np.cos(hoehe_h) * np.sin(atan_wert)), 3)
z_cart = np.round((radius_schale * np.sin(hoehe_h)), 3)
return (x_cart, y_cart, z_cart)

# Schleife fuer den BSC
for koord_sterne in data_sterne:
hoehe_h = height(phi, c_sterne2, stundenwinkel)
for atan_sterne in data_sterne:
atan_wert = azimut_atan(phi, c_sterne2, stundenwinkel, hoehe_h)

for xyz_cart in hoehe_h, atan_wert:
x_cart, y_cart, z_cart = kart_Koordinaten(hoehe_h, atan_wert)

# Textausgabe

print("Julianisches Datum: ", jd, "\n")
print("t = ", t, "\n")
print("GMST = ", gmst, "\n")
print("LMST bzw Sternzeit am Ort: = ", lmst, "Uhrzeit UTC", utc, "\n")
if (holo_time == ""):
print("Localtime:", stempel)
print("jahr:", jahr, "monat", monat, "tag", tag, "\n")
print("Kontrollstern:", c_sterne1[teststern], c_sterne2[teststern], "\n")
print("Stundenwinkel = ", stundenwinkel[teststern],"Stundenwinkel in Grad:",
    stundenwinkel[teststern]/15, "\n")
print("Hoehe a = ", hoehe_h[teststern], "\n")
print("Azimuth aTan = ", (atan_wert[teststern]) + 180, "\n")
print("Kartesische Koordinaten: x = ", x_cart[teststern], " y = ", y_cart[teststern], "
    z = ", z_cart[teststern])
```

Zur Kontrolle werden wichtige Parameter ausgegeben um die Ergebnisse prüfen zu können. Der Algorithmus folgt der Beschreibung im Abschnitt Berechnungsschritte. Die Daten werden in eine .csv geschrieben, um sie besser auswerten zu können:

Listing 3: Python - Erstellen der output.csv

```
# Daten in Liste schreiben
Spalten = ["Ra", "Dec", "JD", "t", "GMST", "LMST", "Stundenwinkel", "Hoehe", "Azimut"
    , "x_kartesisch", "y_kartesisch", "z_kartesisch"]

with open('liste.csv', 'r') as f:
reader = csv.reader(f, delimiter=';')
rows = list(reader)

for row, stundenwinkel, hoehe_h, atan_wert, x_cart, y_cart, z_cart in zip (rows,
    stundenwinkel, \
hoehe_h, atan_wert, x_cart, y_cart, z_cart):

row.append(jd)
row.append(t)
row.append(gmst)
row.append(lmst)
row.append(stundenwinkel)
row.append(hoehe_h)
row.append(atan_wert + 180)
row.append(x_cart)
row.append(y_cart)
row.append(z_cart)

rows.insert(0, Spalten)

# Daten in die Datei schreiben
with open('output.csv', 'w', newline='') as f:
writer = csv.writer(f, delimiter=';')
writer.writerows(rows)
```

5.3. Prüfung der Berechnungen

Die Ergebnisse der Berechnung lassen sich punktuell für bestimmte Zeitwerte kontrollieren und gesamthaft im Sinne der Sichtbarkeitskurven. Exemplarisch sei dies für die Höhe h für den Stern Alpha Cam aus dem Sternbild Giraffe dargestellt. Die nachfolgende Zeitreihe (Bilder 18 bis 21) ermittelt alle 60 Minuten die Position des Sterns für ein beliebiges Datum. Daraus lässt sich eine Sichtbarkeitskurve plotten, die die Höhe am Himmel im Sinne der Sichtbarkeit über dem Horizont repräsentiert. Der Stern Alpha Cam ist ein zirkumpolarer Stern vom Standpunkt Wien aus gesehen, der nie unter dem Horizont verschwindet, wie die nachfolgenden Kurven auch zeigen, d.h. die Höhe h ist stets $h > 0$. Da sich die Sichtbarkeit im Laufe eines Jahres durch die Bewegung um die Sonne verändert, sind die Sichtbarkeitskurven für vier Tage über das Jahr verteilt, exemplarisch dargestellt.

Die Maxima und Minima für den entsprechenden Zeitpunkt liegen im Bereich der Kontrolldaten und stimmen im Groben überein. Wie in 4.4 erläutert, wird keine hochgenaue Positionsberechnung angestrebt. Für den 1. Oktober 2023 liefert die Funktion eine maximale Höhe von h = 71.6° und eine minimale Höhe von h = 24.6° über dem Horizont, jeweils gerundet. Zur Illustration noch ein Beispiel für einen nicht zirkumpolaren Stern, hier Rigel. Die Sichtbarkeit ist für den 1. März 2023 angegeben und entspricht dem Bild 22. Man erkennt, das der Stern im wesentlichen unter dem Horizont liegt und nur für einen kurzen Zeitrum sichtbar wird mit einem Maximum von h = 33.5° (gerundet). Bei diesen Berechnungen ist die atmosphärische Refraktion nicht berücksichtigt.

Die Werte für den **Azimut** von Alpha Cam werden tabellarisch für einen Tag ermittelt und mit

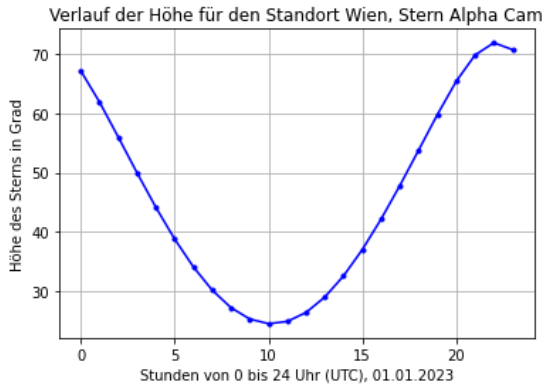

Abbildung 18: Sichtbarkeit Alpha Cam am 1. Jänner 2023

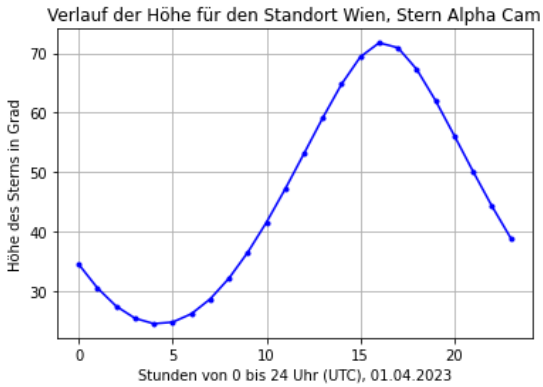

Abbildung 19: Sichtbarkeit Alpha Cam am 1. April 2023

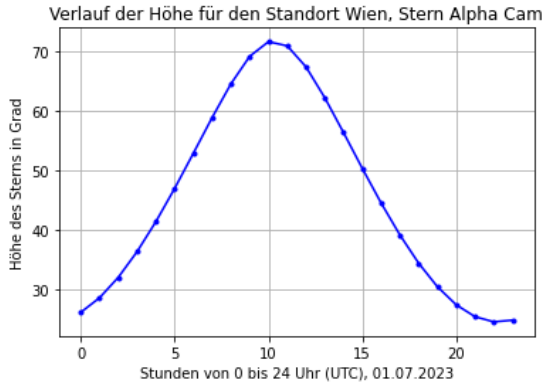

Abbildung 20: Sichtbarkeit Alpha Cam am 1. Juli 2023

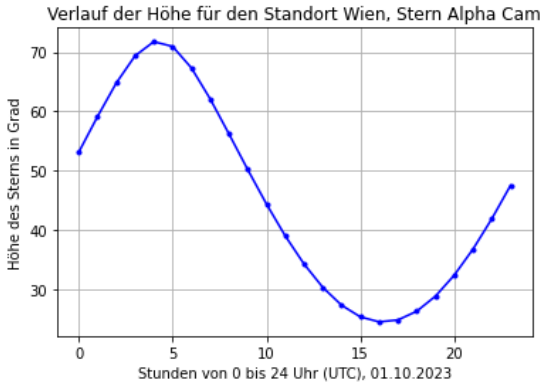

Abbildung 21: Sichtbarkeit Alpha Cam am 1. Oktober 2023

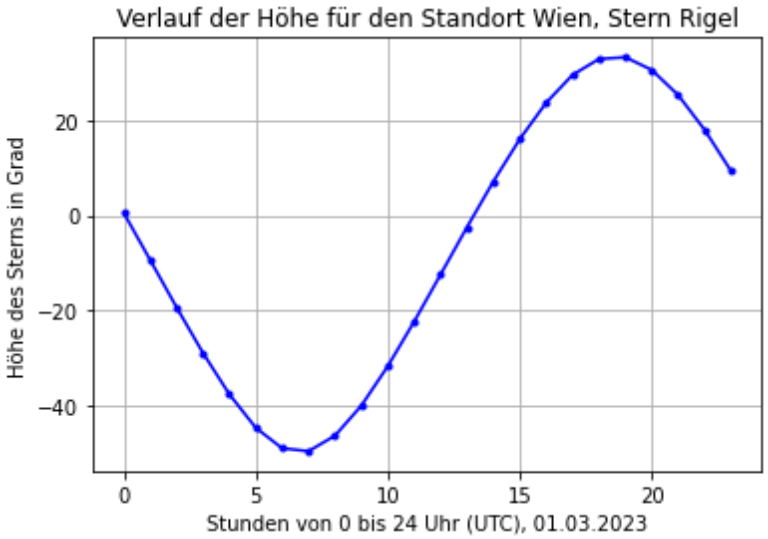

Abbildung 22: Die Kurve zeigt den Verlauf der Höhe h im Horizonsystem für den Stern Rigel. Dieser Stern ist nicht zirkumpolar und man erkennt einen anteilsmäßigen Verlauf unter dem Horizont, d.h. der Stern ist nur teilweise vom Standort Wien sichtbar.

einigen Plots ergänzt; diese benutzen den selben Code als Basis für die Berechnung, da die Höhe h in die Formel für den Azimut benötigt wird. Die berechneten Koordinatenwerte wurden mit den ECO-Tools der NOAJ verglichen. Um Fehlerquellen auszuschließen, werden die Daten für mehrere Sterne berechnet. Die entsprechenden Tabellen sind im Anhang aufgelistet, ergänzend dazu einige Plots und Kennwerte der Daten. Zusätzlich zur stichprobenartigen Kontrolle erfolgt noch eine Prüfung auf Ausreißer im Sinne unsinniger Ergebnisse. Dazu wird die vom Skript erstellte CSV (Auszug zeigt Tabelle 18) in eine SQlite DB importiert und entsprechend auf Fehler hin untersucht. Die Zahlen für die Höhe h müssen entsprechend der Definition zwischen +90° und −90° liegen, die Werte für den Azimut entsprechend der Definition im Bereich zwischen 0° und 360°. Dies wird mit einer einfachen Abfrage (beispielsweise **Select min(height), max(height), min(Azimut), max(Azimut) from output;**) geprüft. Das Ergebnis in Tabelle 11 zeigt, dass die Maxima und Minima im Bereich der Definition liegen und somit keine Ausreißer vorhanden sind. Das schließt auch ein, dass der berechnete Stundenwinkel ebenfalls korrekt ist. Die Werte müssen ebenfalls im Bereich von 0° und 360° liegen. Eine Abfrage aus den Daten zeigt, dass das Minimum bei 0.087° liegt und das Maximum bei 359.943°, also auch im Bereich der sinnvollen Zahlenwerte. Angemerkt sei, dass vorgenannte Maxima und Minima für bestimmte Zeitpunkte stichprobenartig ermittelt wurden.

Um die rechnerische Prüfung einem Praxistest unterziehen zu können, werfen wir noch einen Blick auf den Stern Deneb aus dem Sternbild Schwan, da dieser eine Helligkeit aufweist, die auch mit freiem Auge gut erkennbar ist. Deneb hat die Koordinaten RA 310.35797975 DEC +45.28033881 (IRCS, J2000) und ist somit von Wien aus gut zu sehen. Seine nähere Umgebung ist nicht mit ähnlichen hellen Sternen bestückt, sodass eine visuelle Prüfung der Berechnung möglich ist. Für den Abend des 23. Dezember 2023 um 20 Uhr MEZ (UTC + 1) errechnen wir

Minimum h	Maximum h	Minimum A	Maximum A
−87.796°	88.729°	0.093°	359.996°

Tabelle 11: Die Tabelle listet die Maxima und Minima für Höhe h und Azimut A der Berechnung für alle Sterne aus dem BSC-Katalog auf. Die Daten werden stichprobenartig für bestimmte Zeitpunkte ermittelt.

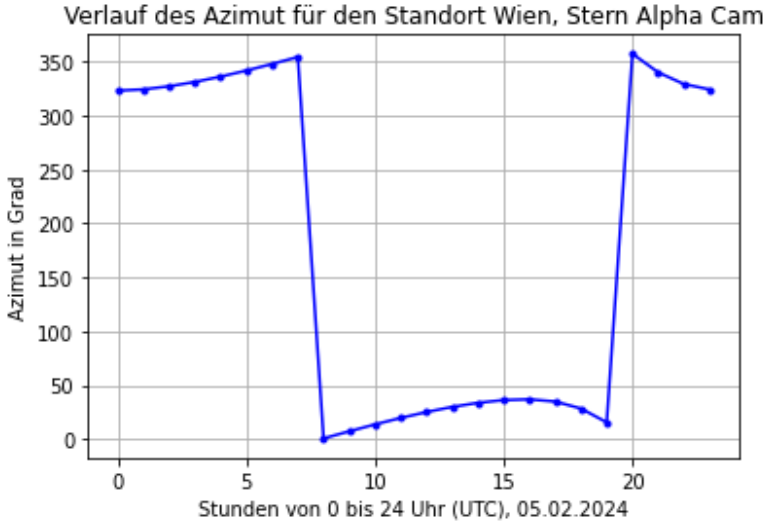

Abbildung 23: Der Verlauf der Werte für den Azimut des Sterns Alpha Camelopardalis für den 5. Februar 2024 von 0 bis 24 Uhr.

einen Azimut von ca 300° sowie eine Höhe von ungefähr 37° (beides gerundet). Für die visuelle Überprüfung sind die Werte ausreichend genau. Um das Ergebnis zu prüfen kann man eine App verwenden, wie man sie zahlreich in den jeweiligen Smartphone App-Stores finden kann bzw. sich mit Kompass und grober Höhenschätzung orientieren. Der Vorteil einer App besteht darin, dass diese durch Verwendung der Sensorik (Position, IMU) und Kalibrierung relativ genau ausgerichtet werden kann, wodurch eine qualitativ gute Prüfung möglich wird. Im konkreten Fall für das Beispiel Deneb stimmen die Werte im Groben überein, woraus auch - unabhängig der errechneten Vergleichswerte im vorigen Abschnitt - von korrekten Zahlen ausgegangen werden kann.

5.4. Visuelle Kontrolle

Abseits der Vergleichsprüfung mit einzelnen Sternkoordinaten dient eine verallgemeinerte Visualisierung der Sterne dazu, das Himmelsbild auch auf seine übliche Erscheinung hin, prüfen zu können. Die optische Plausibilitätsprüfung dient auch als Vorbereitung für die Nutzung in der Hololens.

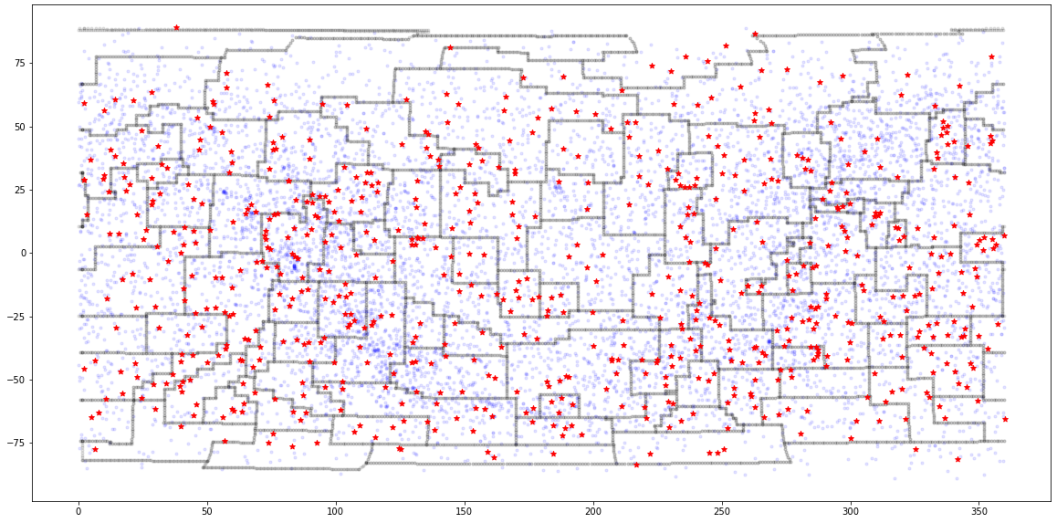

Abbildung 24: Darstellung basierend auf den Werten für Rektaszension und Deklination inkl. Sterne aus dem BSC-Katalog.

5.4.1. Darstellung der Sternbilder

Das Zeichnen der Asterismen wird später direkt in Unity durchgeführt. Die Vorgehensweise auf Basis der Daten für die App wird im Abschnitt Sternbildlinien erläutert. Da für die Zusammenführung beispielsweise eine Gruppierung der Sterne nach Sternbild nötig istm als ein möglicher Lösungsansatz, wird die Gruppierung direkt in Unity über C# und eine LinQ - Abfrage durchgeführt.

5.4.2. Darstellung der Sternbildgrenzen mit allen Sternen

Die Sternbildgrenzen gemäß der Definition der IAU sind aus dem Datensatz VI/49 entnommen und zeigen sich, bei einem Plot wie in Bild Nr 24 ersichtlich. Auf der x-Achse sind die Werte für die Rektaszension aufgetragen, auf der y-Achse jene für die Deklination.

Das Bild 24 beinhaltet aber nicht nur die Grenzen, sondern auch alle Sterne aus dem BSC. Diese sind blau dargestellt, die roten Sterne repräsentieren die Sterne aus dem Asterismen-Katalog von Sluy. Bei näherem Hinsehen kann man auch das für die Milchstraße typische Band erkennen.

5.5. Ablaufdiagramm

Das Diagramm als Bild Nr 25 skizziert den Ablauf der Datenverarbeitung. Zuerst werden das Julianische Datum, die Sternzeit (siderial time) für Greenwich bzw. den Standort ermittelt, dafür benötigt wird die UTC, die aus der entsprechenden Methode der Programmiersprache entnommen wird. Mit diesen Werten kann, unter Einbeziehung der Position der Stundenwinkel bestimmt werden und daraus resultierend die Koordinatentransformation. Im Falle der Testberechnung auf Python-Basis werden die ermittelten Werte in eine .csv - Datei geschrieben, in der

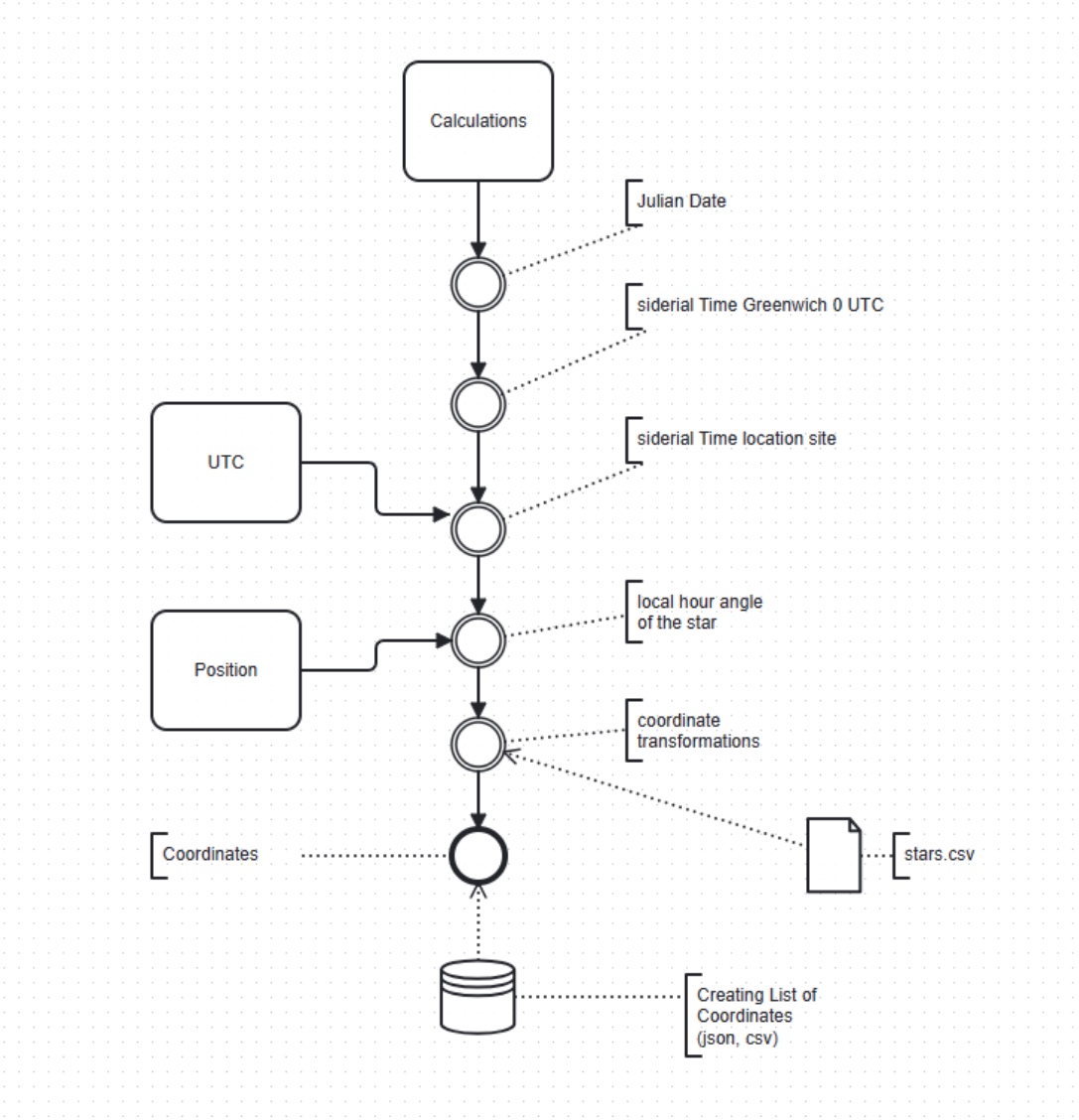

Abbildung 25: Prozess im Sinne der BPMN 2.0 Notation für die Berechnung der Koordinaten

Hololens-Anwendung werden die Daten direkt an Klasse und Methoden übergeben. Das Ablaufdiagramm gilt daher für beide Varianten, also die rechnerische Prüfung mittels Python und die Verwendung in Unity.

6. Anwendungsprogrammierung für die Hololens

In dieser Arbeit wird Unity verwendet, um eine lauffähige Anwendung für die Hololens zu entwickeln. Grundsätzlich gibt es mehrere Möglichkeiten, die Microsoft (Stand Februar 2024) wie folgt auflistet:

- Unity
- Unreal
- native Entwicklung für OpenXR
- Javascript

Unity ist eine für Spieleentwicklung beliebte und häufig genutzte Software, die modular und flexibel einsetzbar ist. In der Unity Umgebung entwickelte Spiele können auf verschiedene Plattformen (Windows, UWP, Android, iOS und andere) portiert werden. Unreal ist ebenfalls eine Game-Engine mit breiter Verwendung, ist jedoch weniger gut mit der Hololens kompatibel. Eine native Entwicklung für den OpenXR Renderer geht im Sinne der Komplexität der Programmierung weit über diese Arbeit hinaus und eine auf Javascript basierende Anwendung erscheint für die Hololens, genutzt im Edge-Browser, nicht die beste Lösung zu sein.

Die Wahl fiel daher auf Unity, weil hierfür die beste Kompatibilität mit der Hololens gegeben und eine weitreichende Dokumentation mit Beispielen vorhanden ist. Ziel dieser Arbeit ist nicht die Erstellung einer komplexen Anwendung, sondern nur die Visualisierung von Sternen im Sinne einer einfachen Demo.

Für die Umsetzung der App gibt es mehrere Ansätze: zum einen kann eine sogenannte Skybox verwendet werden. Damit sind Bilder gemeint, die die Seiten eines Würfels repräsentieren und zusammen gerechnet werden. Dies setzt voraus, dass gerenderte Bilder mit den Sternbildern erzeugt werden müssen. Auf der Überlegungen hinsichtlich der theoretischen Genauigkeit und der formalen Herleitung der Positionen ist diese Möglichkeit keine Option. Ein Beispiel für eine solche App findet sich im Asset Store von Unity [9]. Eine weitere Möglichkeit wäre das Verarbeiten von einem Onlinedienst und Berechnen der notwendigen Koordinaten über eine fremde Schnittstelle (API). Dafür braucht es eine Internetverbindung und entsprechende Dienste sowie eine Integration in die Hololens-Anwendung; die Idee widerspricht teilweise dem Ansatz der Arbeit und wird daher auch nicht weiter verfolgt. Daher bleibt die auf den vorangegangenen Seiten erläuterte Variante: Tabelle mit geeigneten Sternen wählen, Positionen berechnen und in der App darstellen.

[9] https://assetstore.unity.com/packages/3d/environments/sci-fi/real-stars-skybox-plus-151290#content

6.1. Koordinatensystem der Hololens

Die Hololens verwendet ein kartesisches Koordinatensystem mit rechtshändiger Orientierung (Drehsinn gegen den Uhrzeigersinn, mathematisch im positiven Sinne). Das System in Unity ist allerdings links-orientiert, siehe Microsoft (2023a). Es ist daher bei der Entwicklung der Unterschied zu berücksichtigen bzw. die Sternkoordinaten entsprechend umzurechnen. Die ermittelten Koordinaten der Sterne sind in einem sphärischen System mit Azimut A und Höhe h gegeben, d.h. um sie im kartesischen System der Hololens verwenden zu können, müssen sie entsprechend umgerechnet werden. Dies wird auf Basis der Überlegungen im Abschnitt 4.6 besprochen.

6.2. Emulator und Entwicklungsumgebung

Das Testen eines Build über Visual Studio erfolgte mittels Emulator; dieser ist zwar für praktische Tests nur bedingt geeignet, aber zumindest lässt sich die Lauffähigkeit des Programms simulieren und eine Basisansicht zur Prüfung darstellen. Für das MR-Paket gibt es in Unity auch die Möglichkeit des Holographic Remoting[10], diese wurde aber auf Grund der geringen Verfügbarkeit der HL2 nicht benutzt. Für die Tests der Software im Sinne der praktischen Anwendung wurde auf die Geräte der TU Wien, Forschungsgruppe Geoinformation, zurückgegriffen. Die Entwicklungsumgebung im Sinne der Software für die App in der Hololens, Stand Dezember 2023:

1. Microsoft Windows 10 Pro in Version 22H2
2. Visual Studio 2022
3. Unity Version 2021.3.x bzw. 2022.3
4. Hololens Emulator in letztgültiger Version
5. Mixed Reality Tool Kit (MRTK) Version 2.8.3

Die vorgenannte Liste ist die grundsätzliche Umgebung für die Entwicklung der Hololens-App. Zusätzlich kommen für das Testen und die Berechnung der Sternpositionen andere Softwarepakete und Daten zur Anwendung.

6.3. Konfiguration mit MRTK

Die in Unity geschriebene Anwendung wird mittels entsprechender Einstellungen für Unity und Visual Studio auf die Hololens deployed. Dafür wird das von Microsoft zur Verfügung gestellte MRTK verwendet. Mithilfe dieser Sotwareanwendung kann Unity auf effiziente Weise für das Deployment angepasst werden. Es sei an dieser Stelle angemerkt, dass die Konfiguration, die auf Basis der Empfehlungen von Microsoft erfolgt, auch manuell durchgeführt werden kann und vermutlich nicht die einzige mögliche Lösung sein wird.

[10] https://learn.microsoft.com/de-de/windows/mixed-reality/mrtk-unity/mrtk2/features/tools/holographic-remoting?view=mrtkunity-2022-05

Um Unity zu konfigurieren, wird das Mixed Reality Feature Tool (MRFT) benutzt. Die für diese Arbeit verwendete Version ist 1.0.2209.0. Die Konfiguration des Projekts erfolgt gemäß den Hinweisen zur Bedienung des Tools Microsoft (2023c), die nachfolgende Auflistung zeigt die einzelnen Arbeitsschritte:

1. Konfigurieren der Build-Einstellungen
2. Import der nötigen Pakete mit dem MRFT
3. Konfigurieren des Projektes für die Verwendung in der Hololens
4. Testen der Anwendung auf dem Emulator
5. Testen der Anwendung auf der Hololens

Schritt 1 Die Einstellungen müssen für die Hololens dahingehend angepasst werden, da das Programm auf einer anderen Architektur (ARM64) lauffähig sein muss. Die notwendigen Einstellungen zeigt das Bild 27. Damit wird das Unity Projekt für die Nutzung auf der Hololens umgebaut.

Schritt 2 Mit dem MRFT werden nun die notwendigen Komponenten hinzugefügt. Dies sind für die Basisfunktionalität die **Mixed Reality Toolkit Foundation** im Reiter *Mixed Reality Toolkit* sowie im Reiter *Plattform Support* das Paket **Mixed Reality OpenXR Plugin**. Man wählt *GetFeatures* und prozessiert gemäß Software die Installation der Pakete durch. Nach einem Neustart des Unity - Programms sind die entsprechenden Pakete vorhanden und die App kann fertig konfiguriert werden. Im Menüband von Unity wird entsprechend der importierten Pakete ein ergänzendes Menü für die Konfiguration namens **Mixed Reality** angezeigt. Dieser neue Eintrag enthält diverse Einstellungsmöglichkeiten für die Konfiguration des Projekts.

Schritt 3 Nach dem Neustart schlägt die Anwendung verschiedene Basiskonfigurationen vor; es ist die Version für OpenXR zu wählen. Anschließend sind ein paar bestimmte Einträge manuell zu setzen. Die Einstellungen werden im Menü *Project Settings*, Reiter *XR Plugin Management* getätigt.

- im Reiter *UWP* muss **Initialze on start up** aktiv sein
- als Plugin - Provider wird **OpenXR** gewählt
- für OpenXR wird die **Microsoft Hololens FeatureGroup** aktiviert

Diese Einstellungen erzeugen Inkompatibilitäten, die beseitigt werden müssen (die Software zeigt dies durch gelbe Warndreiecke an). Der Reiter *Project Validation* zeigt eine Reihe von Unstimmigkeiten, die die Software automatisch korrigiert. Man wählt für den Reiter *Universal Windows Platform (UWP)* den Modus *fix all*, womit alle Probleme, bis auf das Interaktionsprofil, korrigiert werden (siehe Bild 26). Nach diesen Schritten wird noch das eigentliche Setting hergestellt: man fügt eine entsprechende Szene über das MRTK hinzu, die vorkonfiguriert die

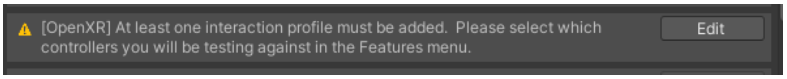

Abbildung 26: Hinweismeldung des MRTK zum Interaktionsprofil. Dieses Problem wird entsprechend den Empfehlungen ausgebessert. Quelle: Screenshot

Parameter für die Verwendung im MR-Universum einstellt. Das Interaktionsprofil wird manuell im Reiter *Open XR* wie folgt ergänzt:

- Eye Gaze Interaction
- Microsoft Hand Interaction Profile
- Microsoft Motion Controller Profile

Das Eye Gaze Interaction Profil erzeugt wieder eine Inkompatibilität und muss, analog zum vorher skizzierten Procedere, beseitigt werden. Danach sind die entsprechenden Profile ergänzt. Abschließend müssen im Reiter OpenXR im Block Feature Groups für die Hololens die Häkchen bei *Hand Tracking* und *Motion Controller* aktiv sein. Der **Deph Submission Mode** wird auf 16 bit gesetzt. Zur leichteren Erkennbarkeit der App kann man im Menü Player unter Project settings den Package Name mit einem wiedererkennbaren Begriff ausstatten. Damit wäre die Konfiguration abgeschlossen.

Schritt 4 Zum Testen auf dem Emulator erzeugt man eine Build-Version. Das erstellte Visual Studio Solution file öffnet man mit Visual Studio und startet das Debugging für den Emulator. Dadurch wird der Emulator in Windows aktiviert und die erzeugte Anwendung wird unmittelbar aufgerufen. Damit der Emulator in Windows genutzt werden kann, sind bestimmte Voraussetzungen notwendig, die ein gewisses Maß an Rechner- und Softwareausstattung einfordern. Beispielsweise kann der Emulator nur mit bestimmten Windows-Versionen genutzt werden und setzt die Virtualisierungsunterstützung voraus. Alle Erfordernisse finden sich aufgelistet auf der Microsoft-Homepage[11], Stand März 2024.

Schritt 5 Um die Anwendung auf die Hololens zu übertragen gibt es verschiedene Möglichkeiten. Das Pairing kann über USB erfolgen, alternativ steht auch WLAN zur Verfügung. Die Anwendung wird auf die Hololens portiert und erscheint danach in der Liste der Apps.

6.4. Positionsbestimmung für die Anwendung

Wie im Abschnitt Grundlagen angeführt, verfügt die Hololens über keine GNSS-Einheit. Da die Position des Anwenders auf der Erde für die korrekte Darstellung der Sternposition aber notwendig ist, muss die Position auf anderem Wege in die Anwendung übermittelt werden.

Grundsätzlich sind mehrere Möglichkeiten in Betracht zu ziehen:

[11] https://learn.microsoft.com/de-de/windows/mixed-reality/develop/advanced-concepts/using-the-hololens-emulator

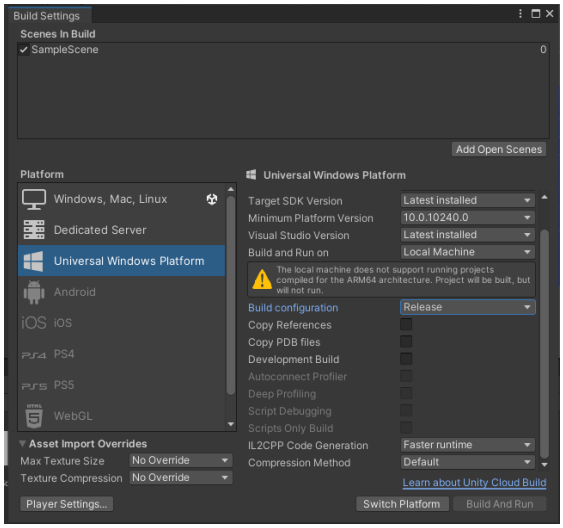

Abbildung 27: Screenshot der Konfiguration für die Build-Einstellungen in Unity

1. vordefiniert für den Demonstrator
2. Textfeld zur manuellen Eingabe der Koordinaten / Auswahlliste
3. Übertragung der Position mittels Smartphone (Bluetooth)
4. Übertragung der Position mittels alternativen GPS-Empfänger / D-GPS

Für den Demonstrator wurden die Koordinaten des Standorts vordefiniert, d.h. die Variablen für ϕ und λ werden im Skript festgelegt und zur Berechnung übergeben. Der Standort wird für Wien mit dem Längengrad 16,36 Ost bzw. dem Breitengrad 48,20 Nord definiert. Die erweiterte Ausbaustufe soll auf Koordinaten zugreifen, die mittels GNSS ermittelt werden.

6.5. Orientierung

Damit die Sterne lagerichtig dargestellt werden können, muss die Blickrichtung des Anwenders bestimmt werden. Der ursprüngliche Ansatz für diese Arbeit war die Nutzung der IMU der Hololens: konkret den Magnetometer für die Orientierung im Sinne der Himmelsrichtungen und das Gyroskop bzw. den Beschleunigungssensor für die Bewegung nach oben oder unten. Die Bewegung zur Seite oder nach oben bzw. nach unten wird aber durch die Verwendung der Sterne als den Benutzer umhüllende Kugelschale nicht benötigt. Die Anwendung sollte unabhängig einer großen Messvorrichtung funktionieren. Um die IMU der Hololens auslesen zu können, muss man den **Research Mode** verwenden. Das Auslesen der Messwerte ist nur mit der HL2 möglich, die Vorgängerversion bietet dieses Feature nicht. Der Research Mode ist im Einstellungsmenü zu aktivieren und nachfolgend im Device Portal (Zugriff via USB, WLAN) noch zusätzlich zu aktivieren. Erst danach kann auf den Datenstrom der diversen Kameras und Sensoren zugegriffen werden.

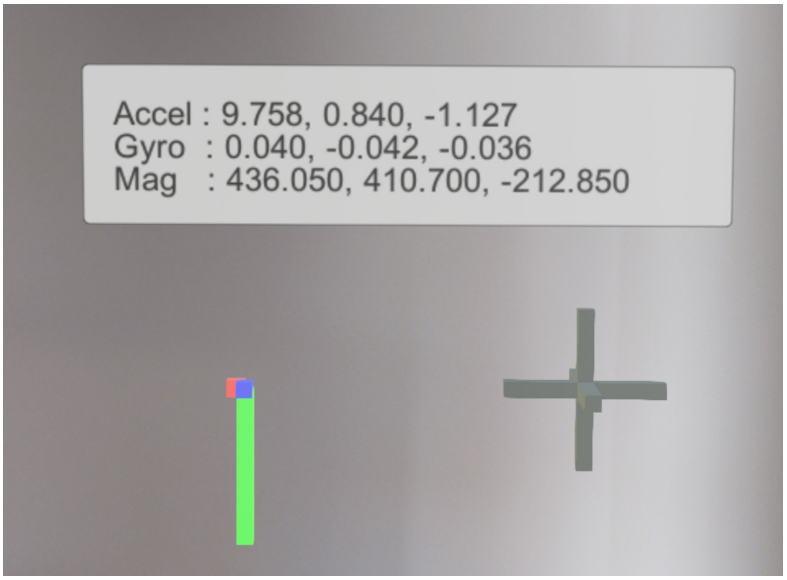

Abbildung 28: Auswertung und Design auf Basis eines Skripts (github) zur Visualisierung der Daten aus dem Sensorpakte der IMU.

Für den Zugriff auf den Datenstrom wurde für diese Arbeit ein Skript auf Basis der von Microsoft zur Verfügung gestellten Information für den Research Mode verwendet. Der entsprechende Quellcode ist auf github[12] verfügbar. Zwar ist diese Variante noch für ältere Unity-Versionen bzw. einem MRTK - Vorgänger konzipiert, aber die grundlegenden Skripte und Treiber (.dll) zum Abgreifen der Daten können 1:1 verwendet werden. Die entsprechenden Codeblöcke für das Auslesen der IMU und die Konfigurationsparameter (manifest.xml) sind im Anhang zitiert. Diese beiden Skripte in Verbindung mit einer Canvas-Vorlage liefern sodann ein Ergebnis wie in Bild 28 gezeigt. Das Bild zeigt die Werte für den Beschleunigungssensor, das Gyroskop sowie den Magnetometer an: Accel steht für den Beschleunigungssensor, Gyro für da Gyroskop und Mag sind die Werte des Magnetometers. Es werden 3 Achsen gemessen, daher liefert die Sensorik 9 Freiheitsgrade.

Da für die Orientierung nur der Magnetometer verwendet werden soll, beschränkt sich der Fokus auf diesen Sensor. Der Magnetometer liefert Messwerte für drei Achsen, die in Abhängigkeit der örtlichen Einflüsse (Magnetfelder in Innenräume durch elektrische Leitungen oder Geräte sowie andere Faktoren) entsprechend stark schwanken. Es hat sich im Zuge der Tests gezeigt, dass die Werte nicht ohne weiteres übernommen werden können, d.h. absolute Werte, die deutlich die magnetische Nordrichtung anzeigen, konnten nicht ermittelt werden; hier simpel als Kompass gedacht. Hierfür wäre eine Kalibrierung nötig und das Ausgleichen der Schwankungen in der gemessenen Flussdichte $\vec{B}$. Ein Beispiel: beim Messen in Innenräumen sind die Werte stark alternierend, wenn man sich in der Nähe von elektrischen Leitungen befindet und darauf zu- oder wegbewegt. Die Werte (eine Achse) reichen hier, bei de facto gleichen Orientierung - von ca 420 bis 450 nT, wenn man nur ein, zwei Schritte vorwärts geht. Daraus resultierend kann man kei-

[12] https://github.com/petergu684/HoloLens2-ResearchMode-Unity

ne einfache Nordrichtung ohne erheblichen Aufwand ableiten. Für diese Anwendung, die auch bei Bewegungen des Users, d.h. Positionsänderungen am Boden, die Nordrichtung ausgleicht, war eine gesicherte Orientierung nicht bestimmbar. Um das Problem der Kompasslösung auszugleichen, bietet sich folgender Ansatz: die Kameraperspektive wird in Unity auf eine Richtung festgelegt, die dem Azimut bei 0° entspricht. Startet der User die App mit Blickrichtung Nord, wird das Abbild der Himmelssphäre näherungsweise nach Nord orientiert. Dieser Ansatz ist naturgemäß mit Ungenauigkeiten behaftet, liefert aber zumindest im Groben eine Position, die ungefähr jener der Sterne entspricht. Da die Hololens die Sterne durch das Erfassen der Umgebung fixiert, wird so eine quasi stabile Verortung ermöglicht, die beim Drehen und Neigen des Kopfes in seiner ursprünglichen Lage stabil bleibt.

6.6. Umsetzung der Anwendung

In Unity werden sogenannte GameObjects verwendet, die unterschiedliche Objekte in 3D-Form repräsentieren. Wir verwenden hier einfache Kugeln, die unsere Sterne darstellen. Es besteht weiters die Möglichkeit, fertige Grafiken in bestimmten Formaten wie zum Beispiel .fbx zu importieren. Da für die Darstellung aber keine anspruchsvollen Grafiken nötig sind, wird auf diese einfachen GameObjects zurückgegriffen. Der Prozess der Datenverarbeitung ist auf den Diagrammen im Abschnitt 5.5 dargestellt.

6.6.1. Import der Daten

Die Daten für die Sterne und die Tabelle mit den Beschriftungen liegen in Form einer .csv-Datei vor und werden für die Berechnung importiert. Dies erfolgt über einen speziellen Ordner in Unity namens **Streaming Assets**. Dateien, die sich in diesem Ordner befinden, werden beim Kompilieren der Anwendung für die UWP-Umgebung auch physisch übernommen und stehen somit für das Auslesen zur Verfügung. Der Zugriff erfolgt über eine einfache Ergänzung des Pfades mittels *Application.dataPath + /StreamingAssets*, somit ist keine Lösung über die FileAccess-Methoden der UWP-Umgebung nötig. Eine Erläuterung zur Vorgehensweise bietet das Handbuch für Unity, siehe Technologies (2024).

6.6.2. Sterne

Beim Start der Anwendung wird die Koordinatenliste gelesen, die Berechnungen durchgeführt und in eine Liste geschrieben. Diese Liste ist nun Basis für die Koordinaten in der Hololens und wird an die Sterne, die on the fly beim Start der Anwendung erzeugt werden, übergeben. Dafür wird in der Anwendung ein Gameobject erstellt, welches über ein prefab angesprochen werden kann. Das Skript greift auf dieses Gameobject zu und mittels der Methode *instantiate* wird jede Koordinate eines Sterns mit einem Vector3-Koordinatensatz versehen und dargestellt. Zeigt man nur die Sterne der Sternbilder an, dies sind ca 700 Sterne, erscheint eine Sphäre aus entsprechenden Sternen wie in Bild 29. Zu beachten ist, dass die Ansicht die Demo innerhalb von Unity zeigt, d.h. der schwarze Hintergrund stellt sich entsprechend der Konfiguration für die Hololens 2 dar (schwarz ist in der Hololens als transparent zu bewerten).

Von außerhalb zeigt sich eine Sphäre mit den gerechneten Sternen, d.h. von einem (geozentrischen) Mittelpunkt aus entspricht die Darstellung der Vorstellung der Himmelsphäre, siehe

Abbildung 29: Darstellung der Sterne sowie der Horizontebene. Gezeigt werden die Sterne, die nur die Sternbilder betreffen. Die Darstellung verwendet als Grafik eine Kugel mit Radius 0.5m.

Bild 30. Für beide Bilder gilt, dass die Sternpositionen on the fly auf Basis des aktuellen Datums bzw. Uhrzeit errechnet und für die Hololens in kartesische Koordinaten umgeschrieben werden. Der Effekt soll, wenn man die Hololens trägt, dazu führen, ein gesamthaftes Bild des Sternenhimmels zu erhalten.

6.6.3. Linien

Für die Darstellung der Linien wird ein LineRenderer - Objekt erzeugt, dass die Punkte untereinander verbindet. Diese sollten in Reihenfolge Anfang - Ende sortiert sein und Punkte teilweise doppelt enthalten, sodass die Figuren stimmig dargestellt werden können. Wie im Abschnitt Sternbilder erklärt, gibt es keine festgelegten Formen, die Darstellung ist kulturabhängig. Die Gruppierung erfolgt anhand der Sternbildnamen für die Sternbildlinien und wird über eine LinQ - Abfrage durchgeführt. An dieser Stelle trennt sich die Darstellung in der Entwicklungsumgebung von jener in der HL2, siehe Ergebnisse. Die Darstellung dient innerhalb von Unity zur Kontrolle der Position und grundsätzlichen Ausrichtung, d.h. die Orientierung der Sternbilder kann dadurch besser geprüft werden. In der Anwendung selbst finden diese Linien aber keine Verwendung.

6.6.4. Sternbildgrenzen

Analog zu den Linien der Sternbilder können die Sternbildgrenzen gezeichnet werden. Es gilt hierbei auch, dass die Punkte von astronomisch-sphärischen Koordinaten in rechtwinkelige Koordinaten überführt werden müssen. Der Datensatz enthält über 13.000 Koordinatenpunkte und wird nur für Orientierungszwecke in der Demo genutzt. Es hat sich allerdings gezeigt, dass die Anzahl der Objekte, als sehr kleine Kugeln oder Linienelemente gerechnet, zu viel Leistung erfordert, siehe Abschnitt Performance. Die Sternbildgrenzen sind ursprünglich für die Epoche B1875 festgelegt worden, der Katalog VI49 enthält aber auch Datenwerte für die Epoche J2000.

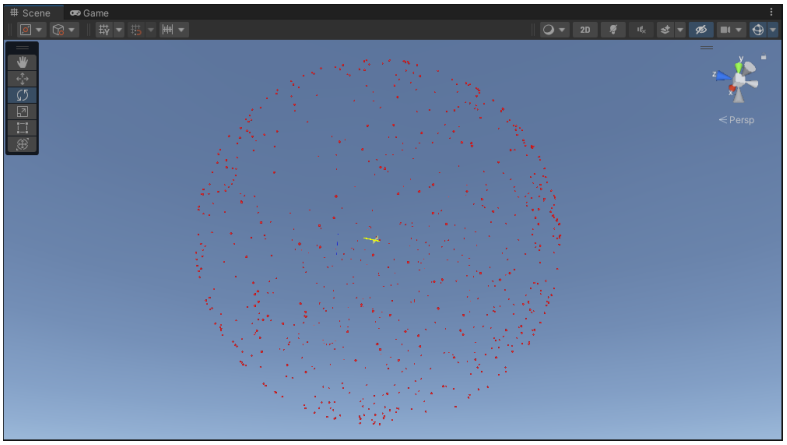

Abbildung 30: Die Sicht zeigt von außen einen Blick auf die gerechnete Sphäre, d.h. alle Sterne der Sternbilder mit x, y. z - Koordinaten, Darstellung über Scene - Ansicht in Unity.

Es ist daher keine Transformation zwischen den Epochen notwendig.

6.6.5. Koordinatennetz

Zur Orientierung und Prüfung der Ausrichtung aller Sterne kann ein Koordinatennetz gezeichnet werden. Dies verwendet ebenfalls GameObjects, in diesem Fall ein Zylinder und wird wieder als Prefab angesteuert. Das Skript errechnet die Linien und markiert den Wert für Längengrad = 0 mit einer anderen Farbe, damit dieser erkennbar ist. Aufgrund der Objektmenge besteht aber das selbe Performance-Problem wie bei den Grenzen oder dem gesamten Sternkatalog, daher wird das Netz beim Testen mit Unity zur Kontrolle verwendet.

6.6.6. Demo mit vordefinierten Koordinaten

Zur Illustration der Anwendung werden vordefinierte Koordinaten genutzt, konkret jene an einem Punkt in Wien mit Längengrad 16,36° Ost und Breitengrad 48,2° Nord. Die Angabe der Ortshöhe (etwa 180 Meter über Meeresspiegel je nach Bezug für die innere Stadt) ist nicht nötig, da wir nicht mit topozentrischen Koordinaten rechnen (siehe Sternposition mit hoher Genauigkeit). Die Koordinaten für den Standort werden entsprechend an die Formeln zur Umrechnung übergeben und die Sterne in das Programm geladen. Die Darstellung des gesamten Himmels vom Standpunkt des Users, die der Kameraperspektive in Unity entspricht, zeigt das Bild 29.

6.6.7. Wenn sich der User bewegt

Die Verortung der Sterne in der Anwendung ist rotationsstabil, d.h. ein Stern an einem bestimmten Punkt bleibt auf diesen Punkt verortet, auch wenn sich der User mit der Hololens dreht. Da die Hololens im Hintergrund permanent die Umgebung erfasst, wird dies ermöglicht. Dies gilt auch für eine Neigungsbewegung, d.h. wenn man den Kopf schräg hält oder nach oben blickt,

bleibt das erzeugte Bild stabil. Diese Verortungsstabilität beschränkt sich allerdings auf einen stehenden Anwender, der sich auf seinem Standpunkt dreht und den Himmel beobachtet. Sobald man sich mit der Hololens bewegt, also sinngemäß seinen Standpunkt verlässt, verliert die Position der Sterne am Himmel seine Gültigkeit, da diese mit dem Anwender mitwandern. Es gilt also die Position um $\triangle x$, $\triangle y$ zu korrigieren. Um diese Positionsveränderung zu erfassen, gäbe es mehrere Möglichkeiten, die man auf zwei Arten durchführen könnte: über Positionsveränderungen durch hardwarebedingte Messungen oder über Lösungen mittels der Hololens Software. Zur Möglichkeit der Messung zählt das Anbinden der Hololens an ein GNSS-System, um die Positionsveränderung des Users zu erfassen. Dies kann via Bluetooth erfolgen, beispielsweise in dem man einem GPS-Empfänger oder ein Smartphone mit entsprechender App verwendet, oder über ein hochgenaues differentielles GNSS-System, wie es an der Forschungsgruppe für Geoinformation der TU Wien verwendet wird. Damit kann man jede Ortsveränderung des Anwenders erfassen und somit die Position neu berechnen. Eine weitere Methode zur Erfassung von $\triangle x$ und $\triangle y$ bestünde in der Verarbeitung aller Sensordaten der IMU. Mit den Werten aus dem Beschleunigungssensor in Verbindung mit dem Gyroskop lassen sich die Bewegung des Benutzers erfassen und entsprechend einem Ineritialsystem auswerten. Da, wie im Abschnitt Orientierung erläutert, das korrekte Verarbeiten der Magnetometerdaten nicht möglich war, fällt diese Option zur Neuausrichtung aus. Grundsätzlich hätte eine Orientierung im Sinne eines Kompass genügt, da die Änderungen in den Koordinaten bei kleinen Werten für $\triangle x$ und $\triangle y$ nicht gravierend sind. Das Gehen von Breitengrad 48.2 auf 48.4 entspricht rechnerisch einer Distanz von ¡ 25 Metern. Diese Positionsveränderung schlägt sich in den Koordinaten für Azimut und Höhe nur hinter der Kommastelle nieder und fällt daher nicht ins Gewicht. Auf Seiten der Softwarelösung könnte eine manuelle Kalibrierung als Ansatz genügen. Um die Orientierung zu fixieren und die Bewegung auszugleichen, die der User am Boden gemacht hat, wäre ein manuelles Ausrichten möglich, sofern man die genaue Nordrichtung kennt. Dies setzt aber ein gewisses Maß an Programmieraufwand voraus, da dadurch alle Sterne entsprechend der Ausrichtung neu verschoben werden müssen.

6.6.8. Build, Deployment

Mit den vorgenannten Features ausgestattet wird die Anwendung für die HL2 erzeugt. Zuvor wird über die Einstellungen in Unity noch der Anzeigename der App angepasst (”Giraffe”) und die App mit einem Splash-Image versehen, welches man bequem über Unity im Menü Project Settings / Player konfigurieren kann. Die Build-Einstellungen wurden im Abschnitt Konfiguration erläutert. Mit diesen Parametern wird ein Build der Anwendung hergestellt, d.h. Unity generiert automatisch alle Dateien und Objekte, die man für das Erzeugen einer Anwendung für die UWP-Umgebung benötigt. Darin enthalten ist auch ein Visual Studio Solution file, welches man mit Visual Studio (VS) öffnet. Die Projektmappenkonfiguration wird auf Release oder Master eingestellt und die Zielplattform gewählt, ARM64 für das Portieren auf die HL2 oder x64 für das lokale Testen über den Hololens-Emulator. Beim Erstellen der Anwendung erzeugt VS die entsprechende .exe - Datei und übermittelt diese an die gewünschte Zielplattform, sodass die erstelle Anwendung automatisch gestartet werden kann.

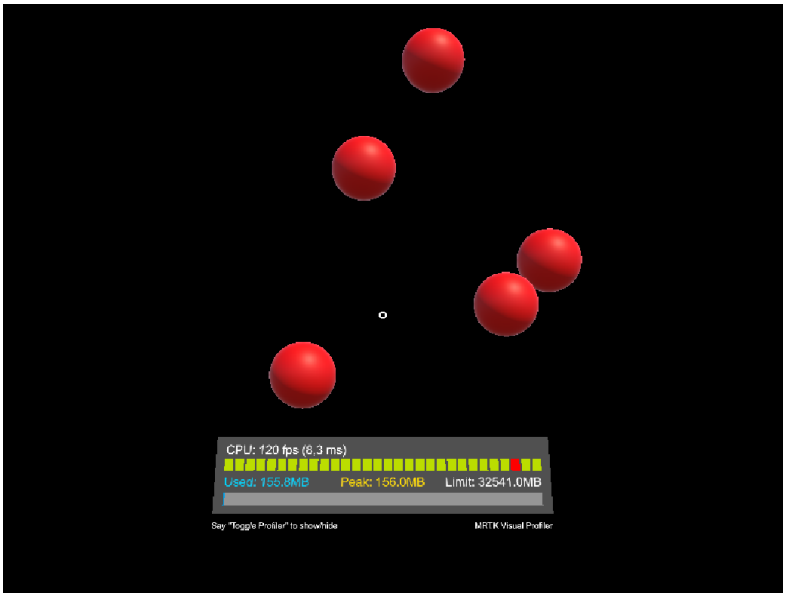

Abbildung 31: Die Darstellung zeigt die berechneten Punktkoordinaten für das Sternbild Giraffe ohne auf die Größe der Sterne Rücksicht zu nehmen, Größe der Sterne stark überzeichnet.

7. Ergebnisse

Die für diese Arbeit erhobenen Genauigkeitsansprüche bezüglich der Koordinaten der Sterne sind durch die einfachen Berechnungen erreicht. Die Sterne als 3D-Objekte in Unity zu projizieren und in die Hololens zu übertragen ist möglich und erfordert, wenn die Datengrundlage passt, nur wenig Programmieraufwand. Ein wichtiger Aspekt ist die Größe der Sterne in der Anwendung selbst. Bei näherungsweise gleicher Größe wie die Sterne am Nachthimmel verringert sich die Erkennbarkeit erheblich, dies zeigt das Bild 32. Bei einer gut sichtbaren Sterngröße generiert die Hololens ein Bild, welches den Nachthimmel ungewöhnlich erscheinen lässt, die Sterne der App aber deutlich sichtbar kennzeichnet. Dies ist durch Bild 33 deutlich zu erkennen.

Für die Anwendung werden etwa 700 Objekte erzeugt und dies führt zu einer relativ hohen Belastung, die sich in einer gewissen Verzögerung bei der Darstellung äußert. Daraus lässt sich für diese Anwendung schlussfolgern, dass die Menge an Objekten, die noch sinnvoll in der Hololens gerechnet werden können, begrenzt ist. Eine mögliche Option zur Verbesserung der Leistung wäre das Filtern aller Sterne, die für den aktuellen Standort nicht sichtbar sind, d.h. alle Sterne mit einem Wert für die Höhe $h < 0$ werden aus der Berechnung gefiltert. Dies reduziert die Anzahl der Objekte erheblich, führt aber auch dazu, dass im Bereich des Horizonts durch Einflüsse der Refraktion ggf. Sterne fehlen würden, die noch sichtbar sind. Aus Gründen der Objektmenge ist auch die Darstellung der Sternbildgrenzen nicht zielführend. Zwar würden die Grenzen zur Orientierung am Himmel beitragen, aber die Tabelle aus dem VI-49 Datensatz enthält über 13.000 Koordinatenpunkte, die selbst bei großzügiger Glättung zu einem erheblichen Berechnungsaufwand führen würden. Dies gilt analog für den BSC-Katalog. Die

Abbildung 32: Die Darstellung zeigt einen Ausschnitt aus der fertigen App. Die gelben Sterne geben die Positionen der Sternbilder (Sternbild-Sterne) wieder. Die grünen Linien stellen ein Gradnetz zur Orientierung dar, der Stern in Pink ist ein Kontrollstern. In dieser Darstellung haben die Sterne einen Radius von 0.001 Meter, sodass sie näherungsweise der Größe der Sterne am Nachthimmel entsprechen.

Abbildung 33: Darstellung der Sterne in der Hololens mit einem Radius von 0.1 Meter. Die Größe der Sterne macht sie gut sichtbar, generiert aber ein ungewohntes Bild.

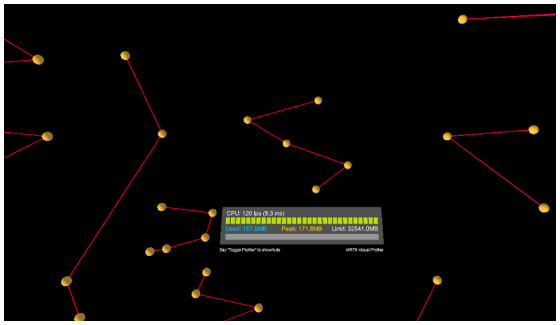

Abbildung 34: Darstellung der Sterne und Verbindungslinien in Unity

Abbildung 35: Darstellung der Sterne und Verbindungslinien in der HL2

zusätzlichen 8400 Sterne, die zur besseren Orientierung und Vervollständigung des Nachthimmels bei guten Lichtverhältnissen einen authentischen Eindruck vermitteln können, übersteigen die Kapazitäten in der für diese Arbeit erstellten Anwendung. Ganz allgemein weisen Microsoft und Unity Technologies in ihren Dokumentationen auf gewisse Einstellungen zur Verbesserung der Performance bzw. Stabilität hin, die man berücksichtigen sollte.

Im Zuge der Entwicklung des Demonstrators konnte festgestellt werden, dass die Unterschiede zwischen dem Preview in Unity und dem tatsächlichen Erscheinungsbild in der Hololens teilweise exorbitant sind. Trotz eines korrekten Profils für die HL2 ist es nicht möglich, bestimmte Objekte oder Methoden aus Unity 1:1 in die Hololens zu übertragen. Dies betrifft weniger die Sterne, die als einfache GameObjects vor allem hinsichtlich der Erscheinungsgröße optimiert werden müssen, sondern viel mehr die Asterismen, die über den Objekttyp LineRenderer erzeugt werden. Die beiden Grafiken zum Vergleich der Unterschiede bei gleicher Konfiguration verdeutlichen dies. In diesem Beispiel haben die Sterne eine Distanz von 5 Metern zum User (Kameraposition in Unity), die Sterne einen Durchmesser von 0.1 Meter und die Linien eine Breite von 0.001 Meter. Bei der Entwicklung des Demonstrators musste also berücksichtigt werden, dass der Code zwischen Entwicklungsumgebung und App zu unterschiedlichen Darstellungen führt. Das Einblenden von Beschriftungen ist ebenfalls mit Schwierigkeiten behaftet. Einerseits durch die korrekte Darstellung und Größe der Schrift (Textobjekte, Canvas-Bereich), welcher beispielsweise als TMP Text oder aus dem MRTK-Fundus erzeugt werden kann. Andererseits müssen die erzeugten Felder dem Blick des Users folgen, d.h. die Ausrichtung zur Kamera hin muss gesichert sein. Durch die Fixierung der Sphäre werden alle Textfelder entgegen der Blickrichtung spiegelverkehrt visualisiert, auch sind die Felder im Bereich des Zenit oder Nadir nicht lesbar, weil keine automatische Neigung zur Blickrichtung erfolgt. Dies kann beispielsweise über eine angepasste rotations-Methode in Unity, angelehnt an die Kameraposition, korrigiert werden.

8. Diskussion

Die Darstellung der Sternbilder im Sinne von einfachen Kugeln, die die Sterne lagerichtig, entsprechend ihre Position am Himmel zum Zeitpunkt der Benutzung der HL2 repräsentieren, ist möglich. Eine rudimentäre Orientierung auf Basis der Blickrichtung *Azimut* = 0 für den Anwender bei gleichzeitiger Kamerafixierung innerhalb der Anwendung, ermöglicht eine grobe Ausrichtung der virtuellen Sphäre.

Die Thematik Performance zeigte Grenzen der Rechenleistung auf und hinsichtlich der Größe der Objekte lässt sich schlussfolgern, dass eine gewisse Mindestgröße über der tatsächlichen Sterngröße sinnvoll ist, um die Objekte erkennen zu können. Und die eingangs auch gestellte Frage nach den Lichtverhältnissen kann man beantworten: je größer die Hologramm-Sterne, desto unproblematischer die Helligkeit. Mit anderen Worten: in einem herkömmlichen Zimmer oder Büro ist auf Grund des bunten Hintergrunds keine Originalgröße zielführend. Diese ist nur bei Nacht unter freiem Himmel, oder in einem dunklen Raum, sinnstiftend.

Für eine verbesserte Anwendung sind in folgenden Teilbereichen Ergänzungen sinnvoll: Genauigkeit der Sternpositionen, Orientierung der Hololens und die Usability der Anwendung. Um die Genauigkeit der dargestellten Sterne zu verbessern, kann man sich der im Abschnitt 4.4 dargestellten Berechnungsmethoden bedienen. Dadurch erreicht man eine noch bessere Position in der Anwendung und eine damit verbessere Überlagerung mit den tatsächlich am Himmel sichtbaren Sternen. Zusätzlich ist eine Prüfung hin auf besondere Konstellationen relevant, die sich zu bestimmten Zeitpunkten ergeben können. Die obere oder die untere Kulmination zählen hiezu, also die maximale Höhe eines Sterns, die er annehmen kann. Für die Orientierung der Hololens und die daraus resultierenden Parameter wie Längengrad und Breitengrad des Anwenders, Zeitkorrekturen und die Orientierung nach Nord kann eine differentielle GPS-Lösung nützlich sein, die an der Forschungsgruppe für Geoinformation der TU Wien Verwendung findet. Diese ermöglicht das Berechnen hochgenauer Werte und somit eine verbessere Ausrichtung der Sterne. Weiters wäre der Zugriff auf die IMU der Hololens hilfreich, um die Bewegung des Users abzufangen (insbesondere, wenn man mit einem großen Maßstab in der Darstellung arbeitet). Letztlich ist die Anwendung für diese Arbeit nur ein Prototyp ohne die Möglichkeiten, die die Hololens böte: Objektmanipulation, Sprachmodul, etc. Beispielsweise könnte man historische Informationen zu den Sternbildern ergänzen, tiefergehende Daten zu den wichtigen Sternen auflisten oder sich die (weitgehend) bekannten Distanzen der Sterne zu Hilfe machen und begehbare 3D-Sternbilder erzeugen. Mit diesen und anderen Informationen lässt sich die Usability und somit das Anwendererlebnis enorm steigern und die App dahingehend verbessern.

9. Zusammenfassung und Ausblick

Der Prototyp zeigt, dass die Darstellung von Sternbildern möglich ist und man hierfür auch keine Grafiken bzw. vorgefertigte Bilder benötigt. Die Objekte können on the fly auf Basis der entsprechenden Daten erzeugt und projiziert werden. Die Verwendung von GameObjects in Unity zur Darstellung der Sterne ist effizient und praktikabel. Das Illustrieren eines Koordinatennetzes, ebenfalls auf Basis von einfachen GameObjects, ermöglicht eine bessere Orientierung innerhalb des erzeugten Bildes. Ebenso können die Sternbildgrenzen auf die selbe weise erzeugt werden, bringen die Hololens aber auf Grund der Vielzahl an einzelnen Objekten an die Leistungsgrenze. Das Darstellen von Beschriftungen und die Herstellung von Linien (Asterismen) erfordert besondere Methoden im Hinblick auf die Eigenheiten der Darstellung in der Hololens. Die unpräzise Orientierung und somit schwer zu erreichende Überlagerung der Sterne zeigt auf, dass eine saubere Ausrichtung ohne zusätzliche Hilfsmittel nicht möglich ist. Um eine präzise und sich überlagernde Stern-Sphäre erzeugen zu können, die auch Positionsveränderungen des Anwenders ausgleichen kann, muss eine Orientierung mit regelmäßiger Nachführung verwendet werden. Eine solche Nachführung kann durch eine GNSS-basierte Positionierung, welche auch die Orientierung nach Nord ermittelt, erreicht werden. Eine weitere Methode, die auch zusammen mit einer GNSS-Messung verwendet werden kann, ist die Interpretation und Verarbeitung der IMU-Sensorik, insbesondere des Magnetometers für eine entsprechende Ausrichtung nach Norden.

Eine weiter entwickelte Anwendung auf Basis dieser App muss daher die Frage der Orientierung lösen. Hinsichtlich der darstellbaren Informationen (Beschriftung, Linien) sind die Anforderungen der Hololens zu berücksichtigen. Die Diskrepanz zwischen Entwicklungsumgebung und Darstellung in der HL2 erfordert zusätzlichen Programmieraufwand bzw. spezielle Techniken, beispielsweise ist es erforderlich, die Beschriftungen nach der Kameraposition auszurichten. Daraus lässt sich schließen, dass komplexe Objekte nicht ohne erheblichen Aufwand und den Spezifika für die Hologrammdarstellung in der HL2, verwendet werden können. Hinsichtlich des Informationsgehalts allgemein sind für eine bessere Usability ergänzende Informationen zu integrieren. Dies können Sensordaten, Kataloginformationen über die Sterne oder besondere Ereignisse sein.

A. Anhang

A.1. Koordinaten des Sternbilds Giraffe

Rektaszension (°)	**Deklination** (°)
05 10 25.6235	56.1648331
05 09 56.3429	52.6655540
04 51 21.6684	52.7196465
03 29 15.1407	52.9366074
03 29 31.6561	55.4362831
03 19 24.9654	55.4596519
03 19 39.2391	57.4593849
03 15 36.2232	57.4684982
03 17 34.9098	68.4662857
03 36 56.8883	68.4214401
03 41 14.0997	77.4163132
03 46 54.2903	77.4025955
03 50 07.3176	80.3986664
05 21 57.3347	80.1478500
05 38 08.6683	85.1239471
08 31 48.8676	84.6103745
08 41 36.6601	86.0975418
14 12 05.5098	85.9308090
14 27 07.8855	79.4449844
13 35 14.2055	79.3629303
13 36 37.6845	76.3638153
09 28 45.8868	81.4677658
09 22 27.7137	72.9741364
08 12 20.6949	73.1383743
08 08 30.9843	59.6433983
07 11 00.7674	59.8037262
07 11 24.3726	61.8031464
06 17 37.7895	61.9641266

Tabelle 12: Koordinatenrahmen für das Sternbild Giraffe. Die Daten stammen aus der Tabelle VI/49 zu den 88 offiziellen Sternbildern. Die Werte geben die Rektaszension bzw. die Deklination wieder, erstere in Form hh:mm:ss, die Deklination in °.

A.2. Vergleiche Datenqualität

Spica		**ECO Tools**			**Eigene Berechnungen**					
Datum	Uhrzeit	Höhe h	Azimut A	t	Höhe h	Azimut	t	delta Azimut	delta h	delta t
2024-02-05	00:00:00	7,100	115,800	19,623	6,598	114,943	19,557	0,857	0,502	0,067
2024-02-05	01:00:00	15,599	128,108	20,626	15,178	127,180	20,560	0,927	0,421	0,067
2024-02-05	02:00:00	22,693	141,811	21,629	22,390	140,792	21,562	1,019	0,302	0,067
2024-02-05	03:00:00	27,790	157,128	22,632	27,647	156,024	22,565	1,104	0,143	0,067
2024-02-05	04:00:00	30,314	173,767	23,634	30,360	172,623	23,568	1,144	-0,047	0,067
2024-02-05	05:00:00	29,911	190,826	0,637	30,155	189,720	0,570	1,106	-0,244	0,067
2024-02-05	06:00:00	26,642	207,173	1,640	27,061	206,170	1,573	1,003	-0,419	0,067
2024-02-05	07:00:00	20,943	222,056	2,643	21,498	221,179	2,576	0,877	-0,555	0,067
2024-02-05	08:00:00	13,414	235,345	3,645	14,064	234,579	3,579	0,766	-0,650	0,067
2024-02-05	09:00:00	4,625	247,350	4,648	5,337	246,662	4,581	0,688	-0,711	0,067
2024-02-05	10:00:00	-4,947	258,584	5,651	-4,201	257,934	5,584	0,650	-0,746	0,067
2024-02-05	11:00:00	-14,904	269,645	6,654	-14,146	268,990	6,587	0,654	-0,758	0,067
2024-02-05	12:00:00	-24,869	281,230	7,656	-24,121	280,520	7,590	0,710	-0,748	0,067
2024-02-05	13:00:00	-34,406	294,209	8,659	-33,698	293,377	8,592	0,832	-0,708	0,067
2024-02-05	14:00:00	-42,911	309,700	9,662	-42,287	308,659	9,595	1,042	-0,623	0,067
2024-02-05	15:00:00	-49,465	328,877	10,665	-48,996	327,544	10,598	1,332	-0,469	0,067
2024-02-05	16:00:00	-52,852	351,877	11,667	-52,623	350,289	11,601	1,588	-0,229	0,067
2024-02-05	17:00:00	-52,141	16,190	12,670	-52,194	14,581	12,603	1,609	0,053	0,067
2024-02-05	18:00:00	-47,551	38,034	13,673	-47,840	36,630	13,606	1,404	0,289	0,067
2024-02-05	19:00:00	-40,216	55,867	14,675	-40,658	54,706	14,609	1,160	0,443	0,067
2024-02-05	20:00:00	-31,279	70,375	15,678	-31,808	69,395	15,612	0,980	0,528	0,067
2024-02-05	21:00:00	-21,542	82,773	16,681	-22,109	81,901	16,614	0,872	0,567	0,067
2024-02-05	22:00:00	-11,536	94,096	17,684	-12,108	93,273	17,617	0,823	0,572	0,067
2024-02-05	23:00:00	-1,667	105,143	18,686	-2,216	104,321	18,620	0,822	0,549	0,067

Tabelle 13: Vergleichstabelle für den Stern Spica

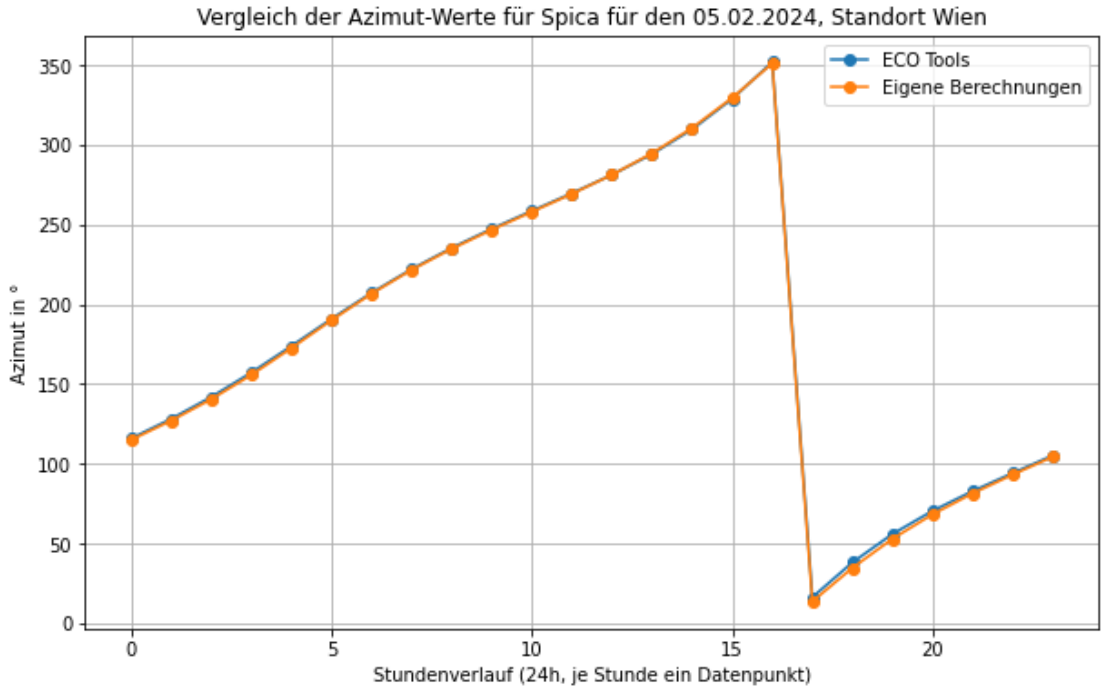

Abbildung 36: Vergleich Azimut A für den Stern Spica

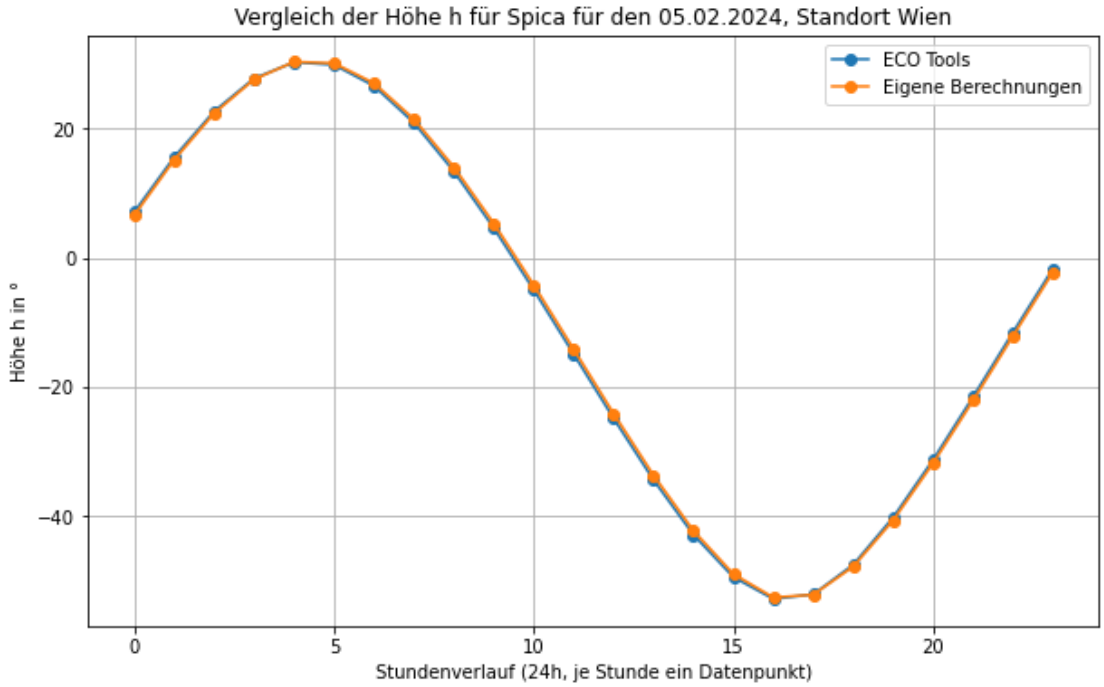

Abbildung 37: Vergleich Höhe h für den Stern Spica

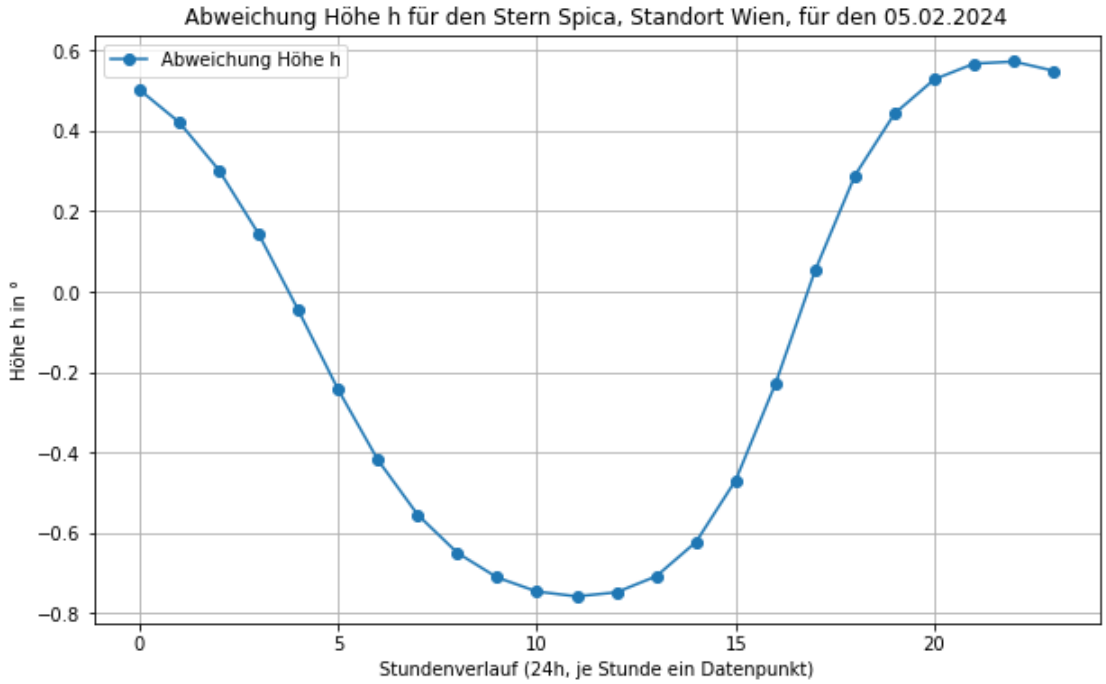

Abbildung 38: Abweichung in der Höhe von der Referenztabelle für Spica

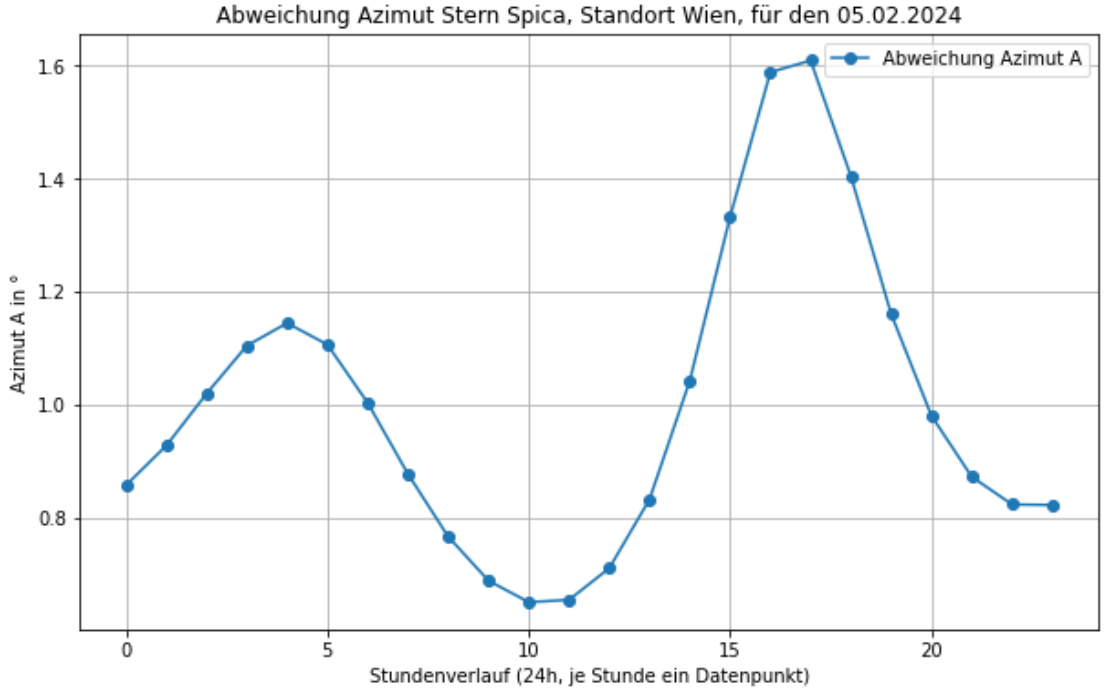

Abbildung 39: Abweichung Azimuth A von der Referenztabelle für Spica

Regulus		**ECO Tools**			**Eigene Berechnungen**					
Datum	Uhrzeit	Höhe h	Azimut A	t	Höhe h	Azimut	t	delta Azimut	delta h	delta t
2024-02-05	00:00:00	51,141	153,786	22,903	50,954	152,258	22,837	1,528	0,187	0,066
2024-02-05	01:00:00	53,629	177,675	23,906	53,712	176,028	23,840	1,646	-0,083	0,066
2024-02-05	02:00:00	51,907	201,956	0,909	52,262	200,472	0,843	1,485	-0,355	0,066
2024-02-05	03:00:00	46,502	223,018	1,912	47,054	221,837	1,845	1,180	-0,552	0,066
2024-02-05	04:00:00	38,650	240,000	2,914	39,317	239,078	2,848	0,922	-0,667	0,066
2024-02-05	05:00:00	29,434	253,896	3,917	30,159	253,138	3,851	0,758	-0,725	0,066
2024-02-05	06:00:00	19,578	265,938	4,920	20,325	265,266	4,854	0,673	-0,747	0,066
2024-02-05	07:00:00	9,572	277,110	5,923	10,317	276,465	5,856	0,645	-0,745	0,066
2024-02-05	08:00:00	-0,196	288,171	6,925	0,525	287,509	6,859	0,662	-0,721	0,066
2024-02-05	09:00:00	-9,346	299,753	7,928	-8,674	299,033	7,862	0,720	-0,672	0,066
2024-02-05	10:00:00	-17,444	312,402	8,931	-16,849	311,588	8,865	0,814	-0,595	0,066
2024-02-05	11:00:00	-23,961	326,525	9,934	-23,481	325,593	9,867	0,932	-0,480	0,066
2024-02-05	12:00:00	-28,303	342,206	10,936	-27,979	341,159	10,870	1,047	-0,324	0,066
2024-02-05	13:00:00	-29,946	358,966	11,939	-29,809	357,848	11,873	1,118	-0,137	0,066
2024-02-05	14:00:00	-28,656	15,794	12,942	-28,713	14,680	12,875	1,114	0,057	0,066
2024-02-05	15:00:00	-24,618	31,645	13,944	-24,848	30,597	13,878	1,047	0,231	0,066
2024-02-05	16:00:00	-18,333	45,963	14,947	-18,701	45,006	14,881	0,957	0,368	0,066
2024-02-05	17:00:00	-10,398	58,774	15,950	-10,864	57,898	15,884	0,876	0,466	0,066
2024-02-05	18:00:00	-1,351	70,458	16,953	-1,882	69,635	16,886	0,823	0,531	0,066
2024-02-05	19:00:00	8,363	81,551	17,955	7,797	80,744	17,889	0,807	0,566	0,066
2024-02-05	20:00:00	18,359	92,672	18,958	17,785	91,838	18,892	0,834	0,574	0,066
2024-02-05	21:00:00	28,256	104,561	19,961	27,704	103,646	19,895	0,915	0,552	0,066
2024-02-05	22:00:00	37,583	118,169	20,964	37,092	117,105	20,897	1,065	0,491	0,066
2024-02-05	23:00:00	45,651	134,708	21,966	45,280	133,416	21,900	1,291	0,372	0,066

Tabelle 14: Vergleichstabelle für den Stern Regulus

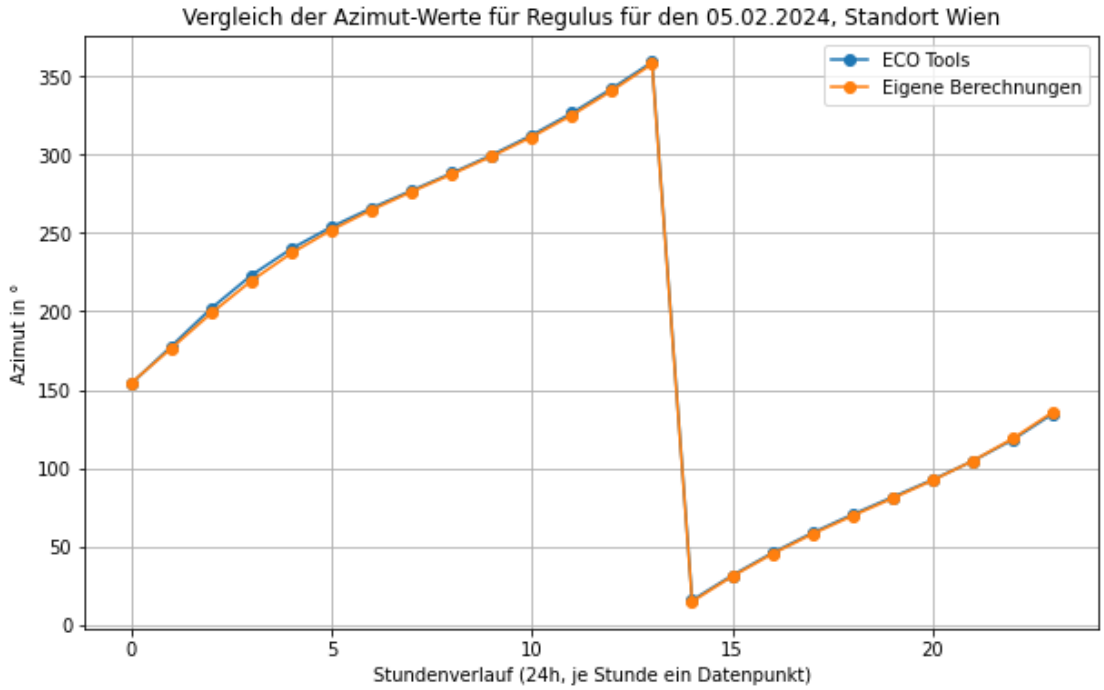

Abbildung 40: Vergleich Azimut A für den Stern Regulus

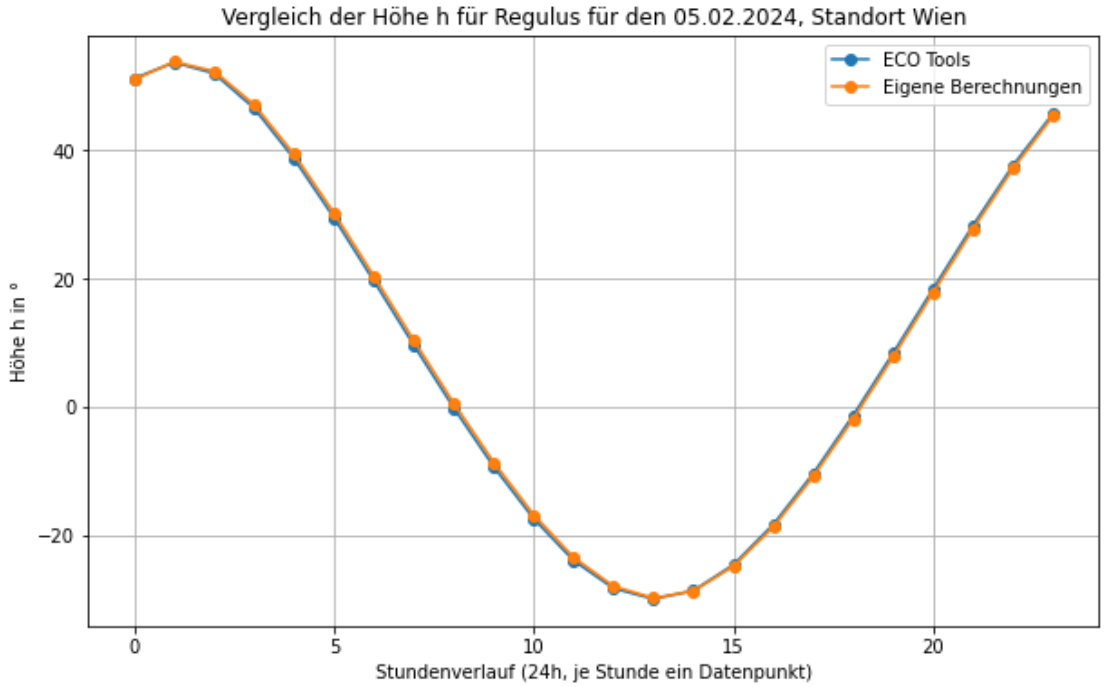

Abbildung 41: Vergleich Höhe h für den Stern Regulus

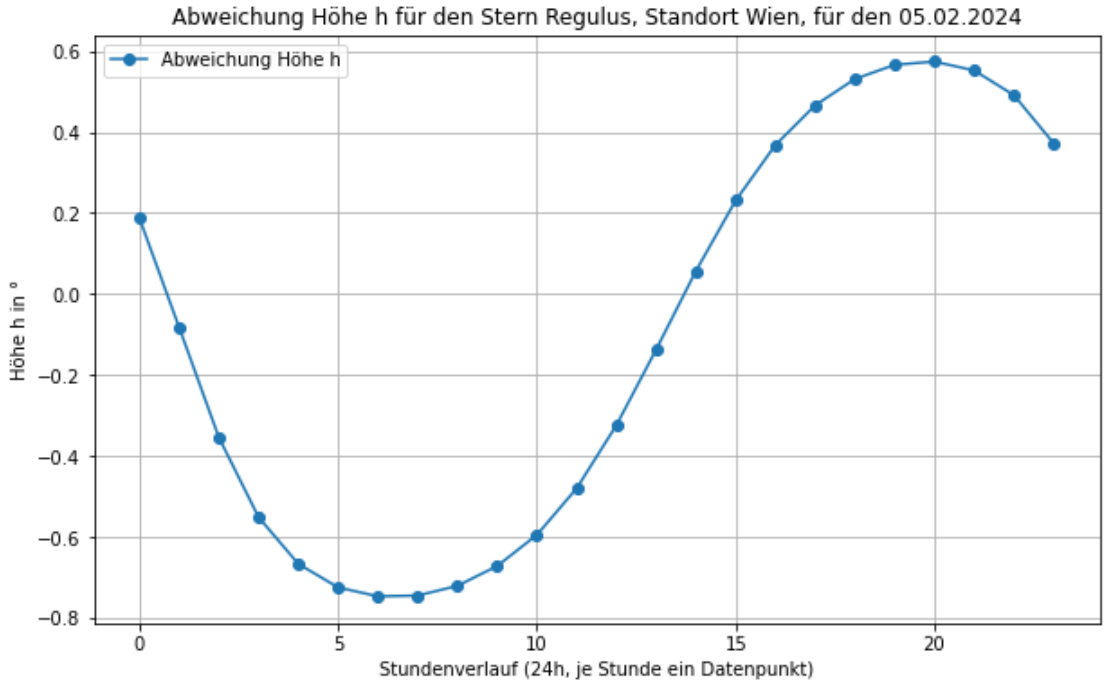

Abbildung 42: Abweichung in der Höhe von der Referenztabelle für Regulus

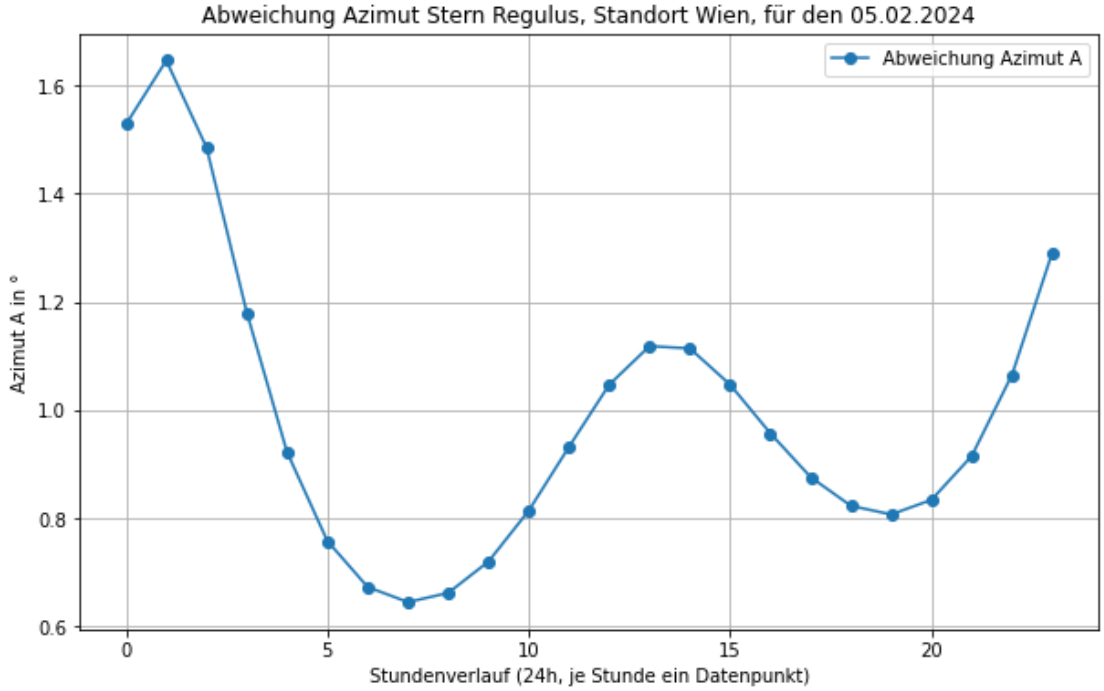

Abbildung 43: Abweichung Azimuth A von der Referenztabelle für Regulus

Rigel		**ECO Tools**			**Eigene Berechnungen**					
Datum	Uhrzeit	Höhe h	Azimut A	t	Höhe h	Azimut	t	delta Azimut	delta h	delta t
2024-02-05	00:00:00	14,650	239,143	3,803	15,213	238,268	3,734	0,875	-0,563	0,068
2024-02-05	01:00:00	5,564	251,136	4,806	6,189	250,331	4,737	0,805	-0,625	0,068
2024-02-05	02:00:00	-4,181	262,405	5,808	-3,523	261,629	5,740	0,776	-0,658	0,068
2024-02-05	03:00:00	-14,184	273,590	6,811	-13,521	272,798	6,743	0,792	-0,663	0,068
2024-02-05	04:00:00	-24,057	285,405	7,814	-23,416	284,544	7,745	0,861	-0,641	0,068
2024-02-05	05:00:00	-33,339	298,710	8,817	-32,757	297,716	8,748	0,994	-0,581	0,068
2024-02-05	06:00:00	-41,386	314,522	9,819	-40,917	313,326	9,751	1,196	-0,470	0,068
2024-02-05	07:00:00	-47,277	333,713	10,822	-46,991	332,280	10,754	1,433	-0,286	0,068
2024-02-05	08:00:00	-49,914	355,956	11,825	-49,882	354,379	11,756	1,576	-0,032	0,068
2024-02-05	09:00:00	-48,621	18,774	12,827	-48,858	17,276	12,759	1,498	0,237	0,068
2024-02-05	10:00:00	-43,747	39,174	13,830	-44,197	37,909	13,762	1,265	0,450	0,068
2024-02-05	11:00:00	-36,334	56,100	14,833	-36,920	55,066	14,764	1,034	0,586	0,068
2024-02-05	12:00:00	-27,398	70,161	15,836	-28,059	69,290	15,767	0,871	0,662	0,068
2024-02-05	13:00:00	-17,673	82,390	16,838	-18,369	81,613	16,770	0,777	0,697	0,068
2024-02-05	14:00:00	-7,669	93,708	17,841	-8,371	92,967	17,773	0,741	0,702	0,068
2024-02-05	15:00:00	2,210	104,872	18,844	1,528	104,119	18,775	0,753	0,682	0,068
2024-02-05	16:00:00	11,575	116,543	19,847	10,941	115,734	19,778	0,809	0,635	0,068
2024-02-05	17:00:00	19,983	129,326	20,849	19,429	128,422	20,781	0,904	0,555	0,068
2024-02-05	18:00:00	26,882	143,724	21,852	26,449	142,696	21,784	1,028	0,433	0,068
2024-02-05	19:00:00	31,622	159,921	22,855	31,356	158,772	22,786	1,149	0,267	0,068
2024-02-05	20:00:00	33,593	177,460	23,858	33,529	176,244	23,789	1,217	0,064	0,068
2024-02-05	21:00:00	32,485	195,192	0,860	32,632	193,999	0,792	1,193	-0,147	0,068
2024-02-05	22:00:00	28,478	211,851	1,863	28,809	210,757	1,795	1,094	-0,331	0,068
2024-02-05	23:00:00	22,124	226,764	2,866	22,595	225,791	2,797	0,973	-0,471	0,068

Tabelle 15: Vergleichstabelle für den Stern Rigel

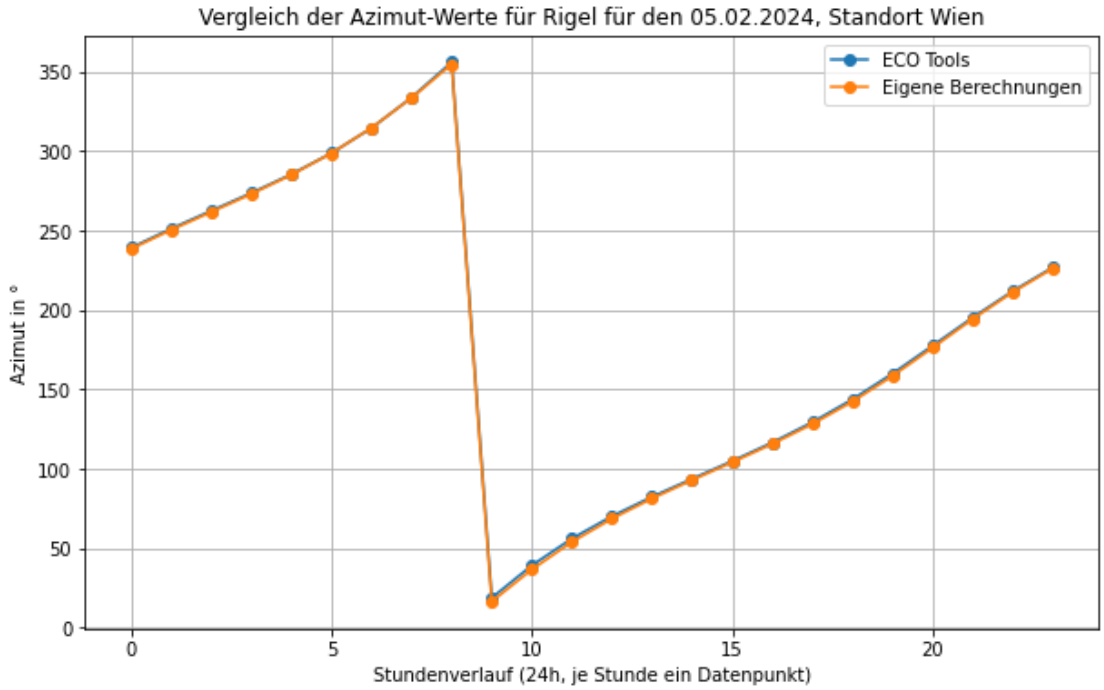

Abbildung 44: Vergleich Azimut A für den Stern Rigel

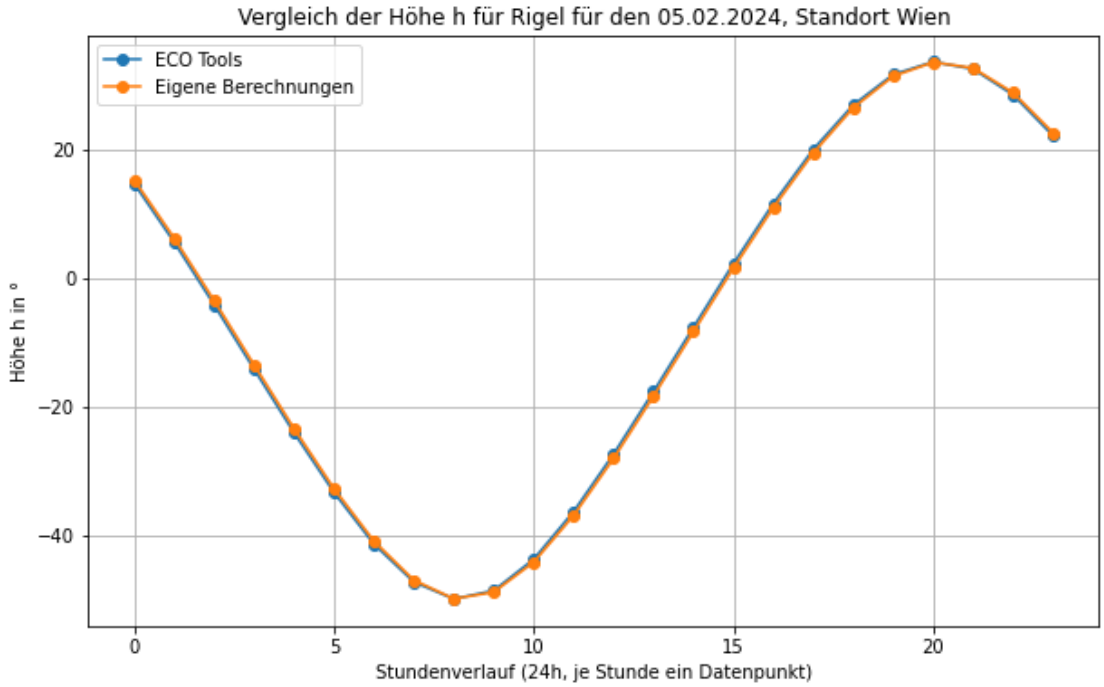

Abbildung 45: Vergleich Höhe h für den Stern Rigel

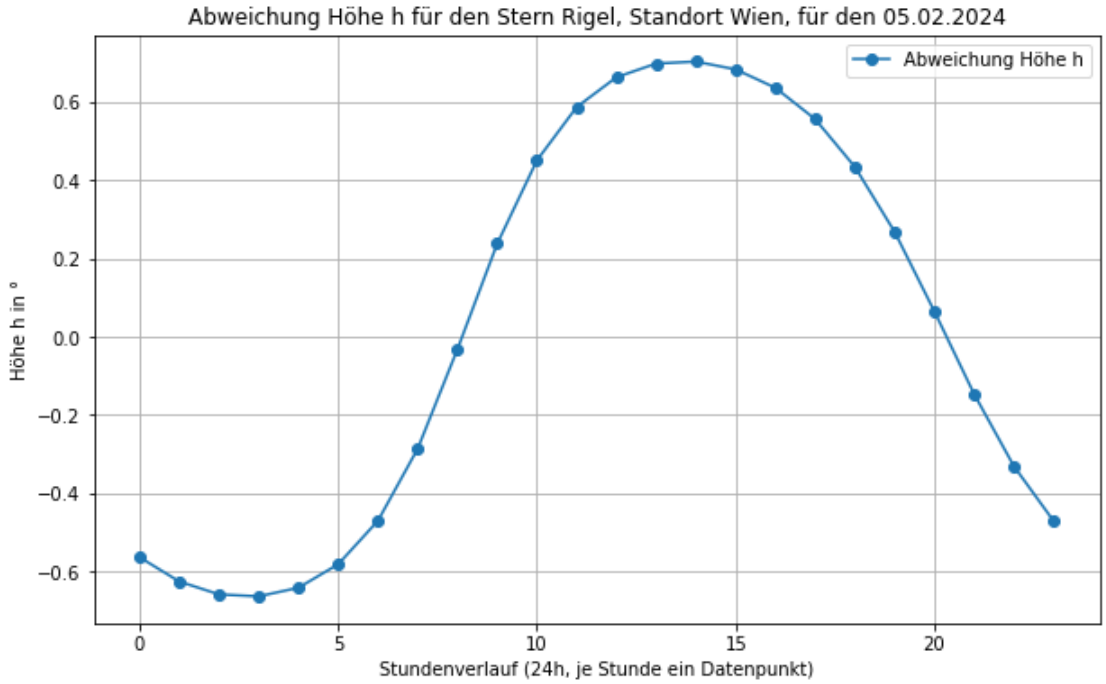

Abbildung 46: Abweichung in der Höhe von der Referenztabelle für Rigel

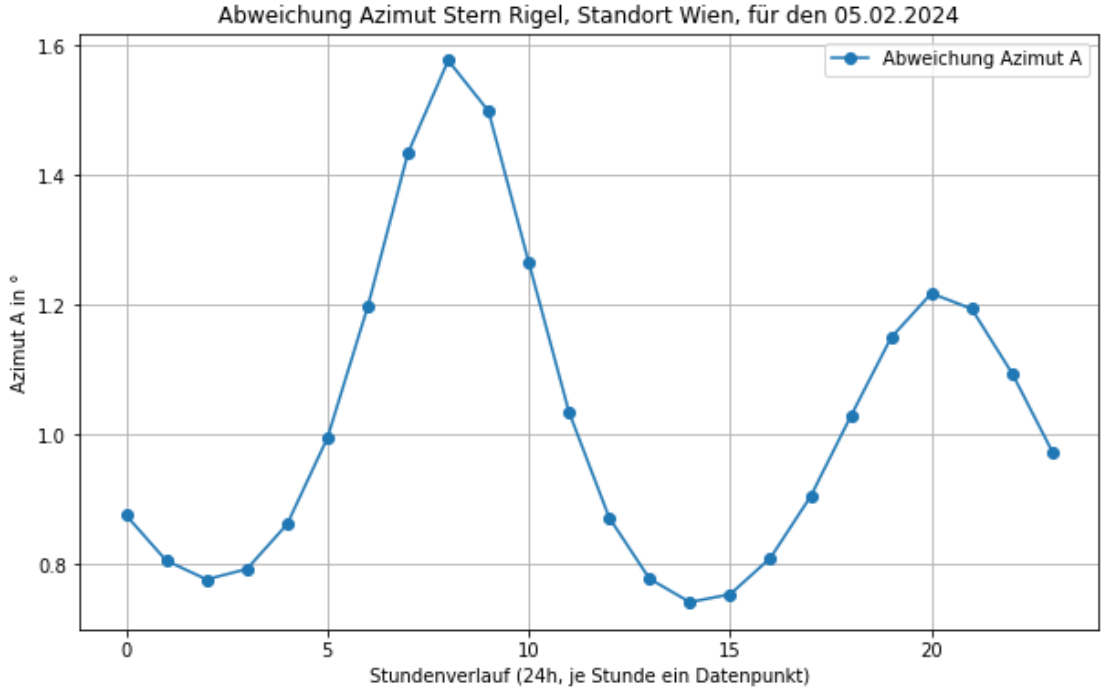

Abbildung 47: Abweichung Azimuth A von der Referenztabelle für Rigel

Alpha Cam		**ECO Tools**			**Eigene Berechnungen**					
Datum	Uhrzeit	Höhe h	Azimut A	t	Höhe h	Azimut	t	delta Azimut	delta h	delta t
2024-02-05	00:00:00	54,0004	323,0666	4,1233	54,2849	322,9868	4,0759	0,0798	-0,2845	0,0474
2024-02-05	01:00:00	48,0401	324,303	5,126	48,3071	324,1396	5,0786	0,1634	-0,2670	0,0474
2024-02-05	02:00:00	42,3809	327,1818	6,1287	42,6201	326,9633	6,0813	0,2185	-0,2392	0,0474
2024-02-05	03:00:00	37,2371	331,2341	7,1315	37,4405	330,9769	7,0841	0,2572	-0,2034	0,0474
2024-02-05	04:00:00	32,7868	336,1702	8,1342	32,9476	335,8852	8,0868	0,2850	-0,1608	0,0474
2024-02-05	05:00:00	29,1838	341,7844	9,137	29,2964	341,4802	9,0895	0,3042	-0,1126	0,0475
2024-02-05	06:00:00	26,5579	347,9021	10,1397	26,6181	347,5867	10,0923	0,3154	-0,0602	0,0474
2024-02-05	07:00:00	25,0089	354,3526	11,1424	25,0143	354,0345	11,0950	0,3181	-0,0054	0,0474
2024-02-05	08:00:00	24,5981	0,9592	12,1452	24,5483	0,6468	12,0978	0,3124	0,0498	0,0474
2024-02-05	09:00:00	25,3422	7,5391	13,1479	25,2391	7,2413	13,1005	0,2978	0,1031	0,0474
2024-02-05	10:00:00	27,2112	13,9112	14,1506	27,0587	13,6361	14,1032	0,2751	0,1525	0,0474
2024-02-05	11:00:00	30,1318	19,9004	15,1534	29,9357	19,6562	15,1060	0,2442	0,1961	0,0474
2024-02-05	12:00:00	33,9946	25,3369	16,1561	33,7621	25,1315	16,1087	0,2054	0,2325	0,0474
2024-02-05	13:00:00	38,6626	30,0408	17,1589	38,4017	29,8834	17,1114	0,1574	0,2609	0,0475
2024-02-05	14:00:00	43,9751	33,7887	18,1616	43,6952	33,6922	18,1142	0,0965	0,2799	0,0474
2024-02-05	15:00:00	49,7458	36,2516	19,1643	49,4576	36,2361	19,1169	0,0155	0,2882	0,0474
2024-02-05	16:00:00	55,7443	36,8807	20,1671	55,4615	36,9803	20,1197	-0,0996	0,2828	0,0474
2024-02-05	17:00:00	61,6486	34,7061	21,1698	61,3912	34,9791	21,1224	-0,2730	0,2574	0,0474
2024-02-05	18:00:00	66,9364	28,0828	22,1726	66,7378	28,6145	22,1251	-0,5317	0,1986	0,0475
2024-02-05	19:00:00	70,7084	15,0555	23,1753	70,6206	15,8853	23,1279	-0,8298	0,0878	0,0474
2024-02-05	20:00:00	71,7609	356,5814	0,178	71,8285	357,4789	0,1306	-0,8975	-0,0676	0,0474
2024-02-05	21:00:00	69,6183	339,5182	1,1808	69,8172	340,1206	1,1333	-0,6024	-0,1989	0,0475
2024-02-05	22:00:00	65,1758	328,9229	2,1835	65,4420	329,1887	2,1361	-0,2658	-0,2662	0,0474
2024-02-05	23:00:00	59,5895	324,1139	3,1862	59,8771	324,1586	3,1388	-0,0447	-0,2876	0,0474

Tabelle 16: Vergleichstabelle für den Stern Alpha Camelopardalis

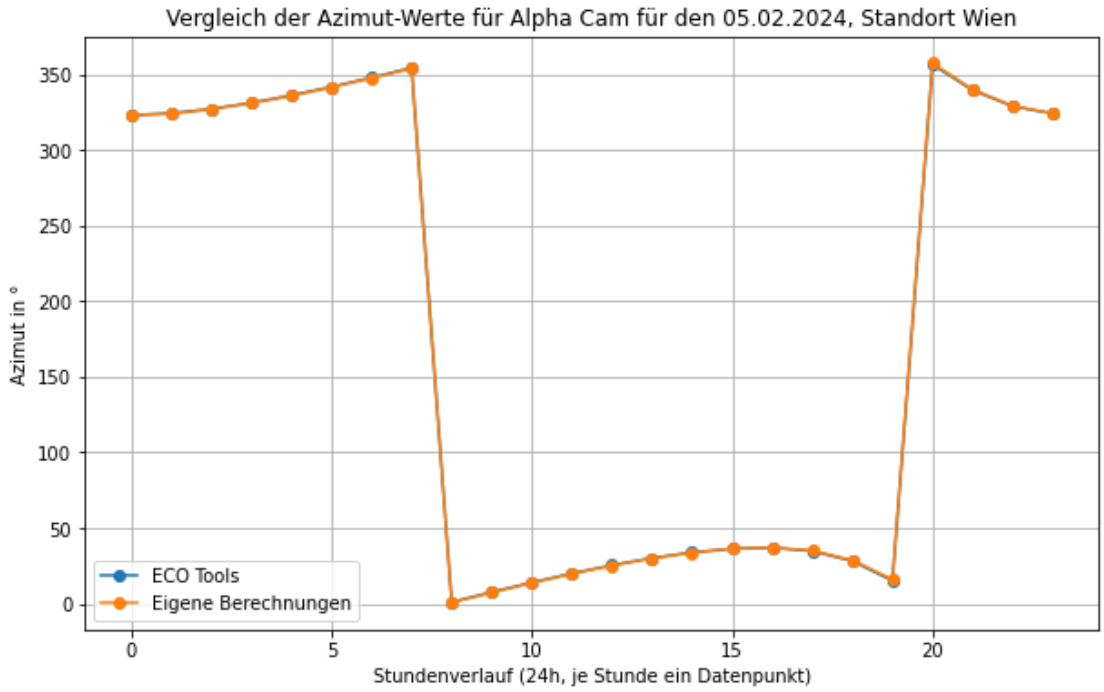

Abbildung 48: Vergleich Azimut A für den Stern Alpha Cam

A.3. Ausstattungsdetails Hololens 2

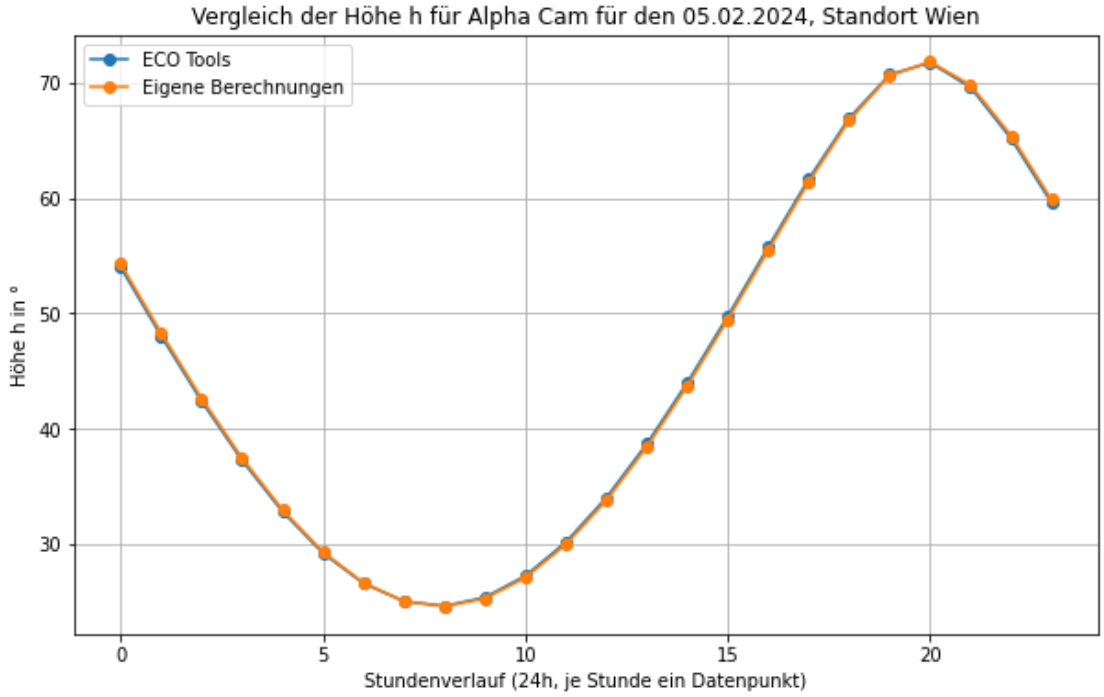

Abbildung 49: Vergleich Höhe h für den Stern Alpha Cam

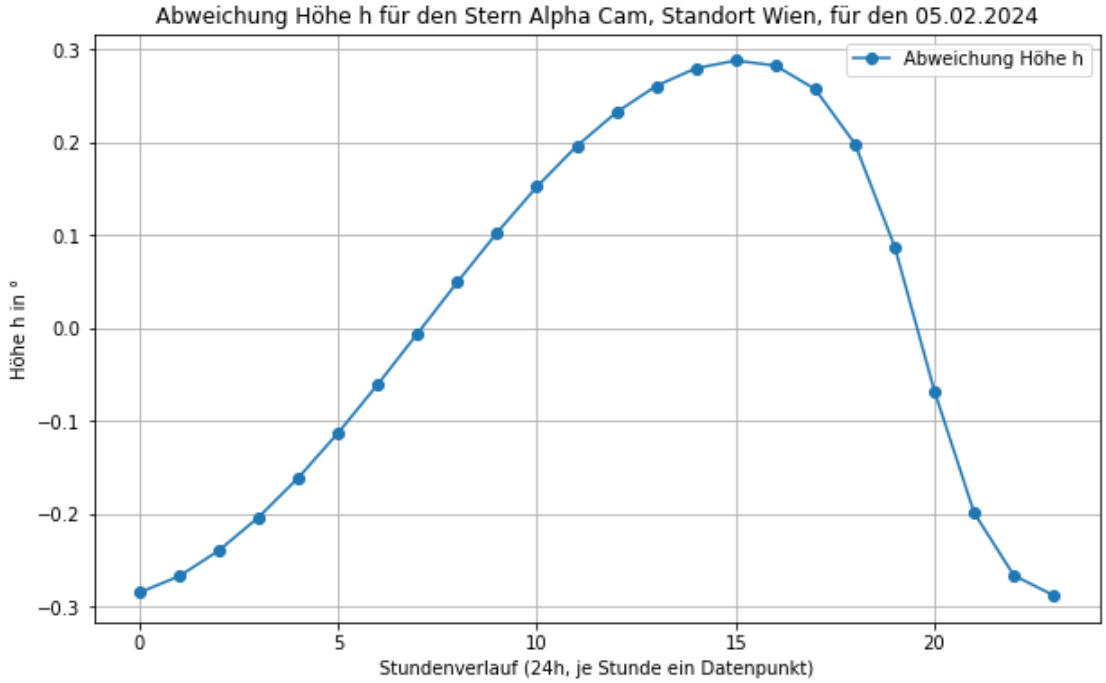

Abbildung 50: Abweichung in der Höhe von der Referenztabelle für Alpha Cam

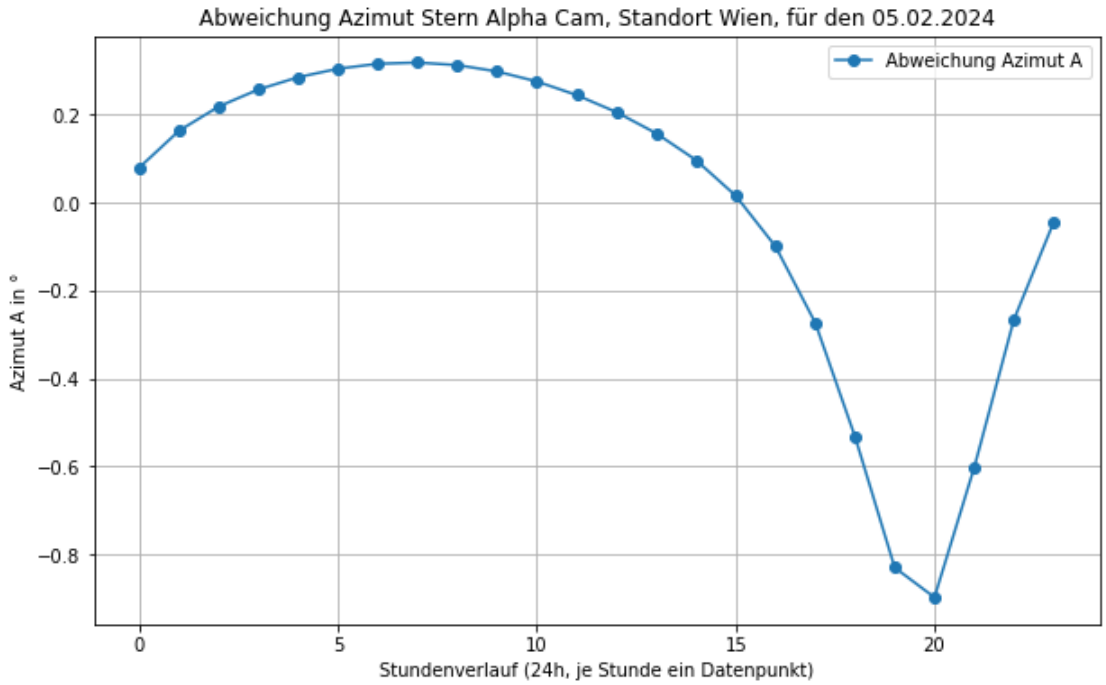

Abbildung 51: Abweichung Azimuth A von der Referenztabelle für Alpha Cam

A.4. Rohdaten output.csv

Display	
Optics	See-through holographic lenses (waveguides)
Resolution	2k 3:2 light engines
Holographic density	>2.5k radiants (light points per radian)
Eye-based rendering	Display optimization for 3D eye position
Sensors	
Head tracking	4 visible light camera
Eye tracking	2 IR cameras
Depth	1-MP time-of-flight (ToF) depth sensor
IMU	Accelerometer, gyroscope, magnetometer
Camera	8-MP stills, 1080p30 video
Audio and speech	
Microphone array	5 channels
Speakers	Built-in spatial sound
Human understanding	
Hand tracking	Two-handed fully articulated model, direct manipulation
Eye tracking	Real-time tracking
Voice	Command and control on-device; natural language with internet connectivity
Windows Hello	Enterprise-grade security with iris recognition
Environment understanding	
6DoF tracking	World-scale positional tracking
Spatial Mapping	Real-time environment mesh
Mixed Reality Capture	Mixed hologram and physical environment photos and video
Compute and connectivity	
SoC	Qualcomm Snapdragon 850 Compute Platform
HPU	Second-generation custom-built holographic processing unit
Memory	4-GB LPDDR4x system DRAM
Storage	64-GB UFS 2.1
Wi-Fi	Wi-Fi: Wi-Fi 5 (802.11ac 2x2)
Bluetooth	5
USB	USB Type-C
Fit	
Single size	Yes
Fits over glasses	Yes
Weight	566g
Software	
Windows Holographic Operating System	
Microsoft Edge	
Dynamics 365 Remote Assist	
Dynamics 365 Guides	
3D Viewer	
Power	
Battery life	2–3 hours of active use
Charging	USB-PD for fast charging
Cooling	Passive (no fans)
Contains lithium batteries	See more information >

Tabelle 17: Die Tabelle listet die technische Ausstattung der Hololens 2 auf. Quelle: Microsoft

Ra	Dec	JD	t	GMST	LMST	Stundenwinkel	Hoehe	Azimut	x_kartesisch	y_kartesisch	z_kartesisch
1.2912	45.2292	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.9281	63.33847	81.59483337402344	-2.304	3.716	2.426
1.2658	-0.5031	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.9535	30.736233	132.87211227416992	-3.889	0.016	-3.143
1.3338	-5.7075	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.8855	26.132738	135.63537216186523	2.506	-1.008	4.207
1.425	13.3961	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.7943	42.465355	123.53347778320312	0.268	0.022	-4.993
1.5667	58.4367	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.6526	64.82249	51.26948547363281	2.032	0.152	4.566
1.5792	-49.075	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.6401	-13.042178	154.75821685791016	4.418	-0.484	-2.29
1.6104	64.1961	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.6089	63.55608	38.33941650390625	-3.59	1.067	3.313
1.6533	29.0214	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.566	54.27405	107.9671630859375	3.155	0.718	-3.812
1.7088	-23.1075	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.5105	10.401995	143.509521484375	-0.991	-2.615	-4.145
1.8258	-17.3864	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.3935	15.503821	140.84962463378906	-0.584	4.861	1.014
1.9338	-2.5489	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.2855	28.624146	133.34707641601562	4.186	2.131	-1.714
1.945	-22.5089	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.2743	10.847615	143.04644012451172	-0.542	-0.5	-4.945
2.0146	-33.5294	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.2047	0.8669514	147.7506103515625	2.175	-2.395	3.812
2.0504	-2.4478	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.16888	28.655914	133.17509078979492	4.435	1.367	-1.862
2.0971	29.0906	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.1222	54.0398	107.42601013183594	3.831	-1.258	-2.957
2.0725	-8.8239	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.1468	23.034092	136.5201759338379	-2.207	-1.213	-4.319
2.1708	36.6267	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.0485	58.73902	96.74807739257812	-0.001	2.903	4.071
2.1392	-17.5775	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.0801	15.200262	140.6591911315918	0.31	4.358	2.431
2.2175	25.4628	2460371.5	0.24165639972621492	160.2771195554176	322.21929191544257	320.0018	51.398815	111.53878784179688	1.681	1.288	4.529

Tabelle 18: Rohdaten output.csv

A.5. Code zum Auslesen der IMU

Listing 4: CSharp - Visualisierung IMU-Daten

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ImuVisualize : MonoBehaviour
{
  public Transform AccelXBar;
  public Transform AccelYBar;
  public Transform AccelZBar;

  public Transform GyroAxisModel;

  public Vector3 AccelVector = Vector3.one;
  public Vector3 GyroEulorAngle = Vector3.zero;

  // Start is called before the first frame update
  void Start()
  {
    if (AccelXBar && AccelYBar && AccelZBar)
    {
      AccelXBar.GetComponent<MeshRenderer>().material.SetColor("_EmissionColor",
          Color.red);
      AccelYBar.GetComponent<MeshRenderer>().material.SetColor("_EmissionColor",
          Color.green);
      AccelZBar.GetComponent<MeshRenderer>().material.SetColor("_EmissionColor",
          Color.blue);
    }

    if (GyroAxisModel)
    {
      foreach (Transform child in GyroAxisModel.transform)
      {
        child.GetComponent<MeshRenderer>().material.SetColor("_EmissionColor", Color.
            white);
      }
    }
  }

  // Update is called once per frame
  void Update()
  {
    if (AccelXBar && AccelYBar && AccelZBar)
    {
      AccelXBar.localScale = new Vector3(Mathf.Abs(AccelVector.x), 0.1f, 0.1f);
      AccelXBar.localPosition = new Vector3(AccelVector.x * 0.5f, 0.0f, 0.0f);

      AccelYBar.localScale = new Vector3(0.1f, Mathf.Abs(AccelVector.y), 0.1f);
      AccelYBar.localPosition = new Vector3(0.0f, AccelVector.y * 0.5f, 0.0f);

      AccelZBar.localScale = new Vector3(0.1f, 0.1f, Mathf.Abs(AccelVector.z));
      AccelZBar.localPosition = new Vector3(0.0f, 0.0f, AccelVector.z * 0.5f);
    }

    if (GyroAxisModel)
    {
      GyroAxisModel.localEulerAngles = new Vector3(GyroEulorAngle.x, GyroEulorAngle.y
          , GyroEulorAngle.z);
    }
```

```
    }
}
```

Listing 5: CSharp - Auslesen der IMU

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using System;
using System.Runtime.InteropServices;

#if ENABLE_WINMD_SUPPORT
using HL2UnityPlugin;
#endif

public class ResearchModeImuStream : MonoBehaviour
{
  #if ENABLE_WINMD_SUPPORT
  HL2ResearchMode researchMode;
  #endif
  private float[] accelSampleData = null;
  private Vector3 accelVector;

  private float[] gyroSampleData = null;
  private Vector3 gyroEulerAngle;

  private float[] magSampleData = null;

  public Text AccelText = null;
  public Text GyroText = null;
  public Text MagText = null;

  public ImuVisualize RefImuVisualize = null;

  void Start()
  {
    #if ENABLE_WINMD_SUPPORT
    researchMode = new HL2ResearchMode();
    researchMode.InitializeAccelSensor();
    researchMode.InitializeGyroSensor();
    researchMode.InitializeMagSensor();

    researchMode.StartAccelSensorLoop();
    researchMode.StartGyroSensorLoop();
    researchMode.StartMagSensorLoop();
    #endif
  }
  void LateUpdate()
  {
    #if ENABLE_WINMD_SUPPORT
    // update Accel Sample
    if (researchMode.AccelSampleUpdated())
    {
      accelSampleData = researchMode.GetAccelSample();
      if (accelSampleData.Length == 3)
      {
        AccelText.text = $"Accel␣:␣{accelSampleData[0]:F3},␣{accelSampleData[1]:F3},␣
            {accelSampleData[2]:F3}";
      }
    }

```

```
57        // update Gyro Sample
58        if (researchMode.GyroSampleUpdated())
59        {
60          gyroSampleData = researchMode.GetGyroSample();
61          if (gyroSampleData.Length == 3)
62          {
63            GyroText.text = $"Gyro␣␣:␣{gyroSampleData[0]:F3},␣{gyroSampleData[1]:F3},␣{
                gyroSampleData[2]:F3}";
64          }
65        }
66
67        // update Mag Sample
68        if (researchMode.MagSampleUpdated())
69        {
70          magSampleData = researchMode.GetMagSample();
71          if (magSampleData.Length == 3)
72          {
73            MagText.text = $"Mag␣␣␣:␣{magSampleData[0]:F3},␣{magSampleData[1]:F3},␣{
                magSampleData[2]:F3}";
74          }
75        }
76        #endif
77        // Convert to Vector3
78        accelVector = CreateAccelVector(accelSampleData);
79        gyroEulerAngle = CreateGyroEulerAngle(gyroSampleData);
80
81        // Visualize corrected values
82        RefImuVisualize.AccelVector = accelVector * 0.1f;
83        RefImuVisualize.GyroEulorAngle = gyroEulerAngle * 30.0f;
84      }
85
86      private Vector3 CreateAccelVector(float[] accelSample)
87      {
88        Vector3 vector = Vector3.zero;
89        if ((accelSample?.Length ?? 0) == 3)
90        {
91          // Positive directions
92          //  accelSample[0] : Down direction
93          //  accelSample[1] : Back direction
94          //  accelSample[2] : Right direction
95          vector = new Vector3(
96          accelSample[2],
97          -1.0f * accelSample[0],
98          -1.0f * accelSample[1]
99          );
100       }
101       return vector;
102     }
103
104     private Vector3 CreateGyroEulerAngle(float[] gyroSample)
105     {
106       Vector3 vector = Vector3.zero;
107       if ((gyroSample?.Length ?? 0) == 3)
108       {
109         // Axis of rotation
110         //  gyroSample[0] : Unity Y axis(Plus)
111         //  gyroSample[1] : Unity Z axis(Plus)
112         //  gyroSample[2] : Unity X axis(Plus)
113         vector = new Vector3(
114         gyroSample[2],
115         gyroSample[0],
116         gyroSample[1]
```

```
        );
      }
      return vector;
    }

    public void StopSensorsEvent()
    {
      #if ENABLE_WINMD_SUPPORT
      researchMode.StopAllSensorDevice();
      #endif
    }

    private void OnApplicationFocus(bool focus)
    {
      if (!focus) StopSensorsEvent();
    }
  }
```

Listing 6: XML - Parameter manifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<Package
xmlns:mp="http://schemas.microsoft.com/appx/2014/phone/manifest"
xmlns:uap="http://schemas.microsoft.com/appx/manifest/uap/windows10"
xmlns:uap2="http://schemas.microsoft.com/appx/manifest/uap/windows10/2"
xmlns:uap3="http://schemas.microsoft.com/appx/manifest/uap/windows10/3"
xmlns:uap4="http://schemas.microsoft.com/appx/manifest/uap/windows10/4"
xmlns:iot="http://schemas.microsoft.com/appx/manifest/iot/windows10"
xmlns:mobile="http://schemas.microsoft.com/appx/manifest/mobile/windows10"
xmlns:rescap="http://schemas.microsoft.com/appx/manifest/foundation/windows10/
    restrictedcapabilities"
IgnorableNamespaces="uap␣uap2␣uap3␣uap4␣mp␣mobile␣iot␣rescap"
xmlns="http://schemas.microsoft.com/appx/manifest/foundation/windows10">
<Identity Name="IMU" Publisher="CN=DefaultCompany" Version="1.0.0.0" />
<mp:PhoneIdentity PhoneProductId="1a2c5f5c-ad36-4c85-81d4-02243efa71ba"
    PhonePublisherId="00000000-0000-0000-0000-000000000000" />
<Properties>
<DisplayName>IMU</DisplayName>
<PublisherDisplayName>DefaultCompany</PublisherDisplayName>
<Logo>Assets\StoreLogo.png</Logo>
</Properties>
<Dependencies>
<TargetDeviceFamily Name="Windows.Universal" MinVersion="10.0.10240.0"
    MaxVersionTested="10.0.22000.0" />
</Dependencies>
<Resources>
<Resource Language="x-generate" />
</Resources>
<Applications>
<Application Id="App" Executable="$targetnametoken$.exe" EntryPoint="IMU.App">
<uap:VisualElements DisplayName="IMU" Square150x150Logo="Assets\Square150x150Logo.png
    " Square44x44Logo="Assets\Square44x44Logo.png" Description="IMU" BackgroundColor
    ="transparent">
<uap:DefaultTile ShortName="IMU" Wide310x150Logo="Assets\Wide310x150Logo.png" />
<uap:SplashScreen Image="Assets\SplashScreen.png" BackgroundColor="#FFFFFF" />
<uap:InitialRotationPreference>
<uap:Rotation Preference="landscape" />
<uap:Rotation Preference="landscapeFlipped" />
<uap:Rotation Preference="portrait" />
<uap:Rotation Preference="portraitFlipped" />
</uap:InitialRotationPreference>
</uap:VisualElements>
```

```
38    </Application>
39    </Applications>
40    <Capabilities>
41    <rescap:Capability Name="perceptionSensorsExperimental" />
42    <Capability Name="internetClient" />
43    <Capability Name="internetClientServer" />
44    <Capability Name="privateNetworkClientServer" />
45    <uap2:Capability Name="spatialPerception" />
46    <DeviceCapability Name="backgroundSpatialPerception"/>
47    <DeviceCapability Name="webcam" />
48    <uap2:Capability Name="spatialPerception" />
49    <DeviceCapability Name="microphone" />
50    <DeviceCapability Name="gazeinput" />
51    <rescap:Capability Name="appCaptureSettings"/>
52    </Capabilities>
53    </Package>
```

A.6. Code in Unity zur Darstellung der Sterne

Listing 7: CSharp - Darstellung Sterne Unity

```
// Skript Sterne.cs, Hildebrandt 03/2024
// Darstellung der Sterne mit Berechnung der Position
// abschnittsweise unter Zuhilfename ChatGPT 3.5, Stand Winter 2023/24

using UnityEngine;
using System.Collections;
using System.Collections.Generic;
using System.IO;
using System;
using UnityEditor;
using TMPro;
using System.Linq;

public class Sterne : MonoBehaviour
{
  // Objekte, die mit Koordinaten versehen werden
  public GameObject Sphere;
  public GameObject Sphere_Demo;
  public GameObject Sphere_Deneb;
  public GameObject Cylinder;
  public GameObject Cylinder_0;
  public LineRenderer lineRendererPrefab;
  public float Linienbreite = 0.01f;

  public float Drehwinkel = 0f;

  // Textfelder
  public TMP_Text Uhrzeit;

  // Koordinaten Wien (Testort)
  public double longitude = 16.36;
  public double latitude = 48.2;

  // Distanz Kamera - Objekte
  public double distance = 5;
  public float faktor = 1;

  // Parameter Koordinatennetz
  public int latitudeLines = 50;
  public int longitudeLines = 50;
  public float radius = 5f;

  void Start()
  {

    // Drehwinkel Kamera
    Camera camera = Camera.main;
    camera.transform.rotation = Quaternion.Euler(0f, Drehwinkel, 0f);

    // Erzeugt Koordinatennetz
    Koordinatennetz();

    // Zeit und Datum
    DateTime currentDateTime = DateTime.Now;
    int year = currentDateTime.Year;
    int month = currentDateTime.Month;
    int day = currentDateTime.Day;
    int hour = currentDateTime.Hour;
```

```
59          int minute = currentDateTime.Minute;
60          int second = currentDateTime.Second;
61          int utc = 0;
62
63          // Berechnen der siderial time und LMST
64          double siderealTime = SiderealTimeCalculator.CalculateSiderealTime(year, month,
                day, utc, longitude);
65          double lmst = siderealTime + ((20 * 15) + (15 * 15 / 60) + ((second * 15) / 3600)
                ) * 1.00273790935;
66          if (lmst < 0)
67          {
68            lmst += 360;
69          }
70          else
71          {
72            lmst -= 360;
73          }
74
75
76                //Uhrzeit und Datum
77          Uhrzeit.text = "Datum:␣" + Convert.ToString(year) + Convert.ToString(month) +
                Convert.ToString(day) + "␣Uhrzeit:␣" + Convert.ToString(hour) + ":" +
                Convert.ToString(minute) + ":" + Convert.ToString(second);
78
79          // Pfad zur CSV
80          string filePath = Application.dataPath + "/StreamingAssets/Constellations_Export.
                csv";
81
82          List<StarData> starDataList = ReadCsvFile(filePath);
83
84          // Linq Abfrage zur Gruppierung nach Sternbild
85          var groupedData = starDataList.GroupBy(s => s.Constellation);
86
87          foreach (var group in groupedData)
88          {
89            List<Vector3> constellationPoints = new List<Vector3>();
90
91            foreach (var star in group)
92            {
93
94              float x_koord = Convert.ToSingle(star.RA);
95              float y_koord = Convert.ToSingle(star.DEC);
96
97              double stundenwinkel = lmst - x_koord;
98              if(stundenwinkel < 0)
99              {
100               stundenwinkel += 360;
101             }
102
103             // Berechnung h
104             double sinH = Math.Sin(Mathf.Deg2Rad * latitude) * Math.Sin(Mathf.Deg2Rad * (
                    y_koord)) +
105             Math.Cos(Mathf.Deg2Rad * latitude) * Math.Cos(Mathf.Deg2Rad * (y_koord)) *
106             Math.Cos(Mathf.Deg2Rad * stundenwinkel);
107             double h = Mathf.Rad2Deg * Math.Asin(sinH);
108
109             // Berechnung A
110             double x_atan = ((-Math.Sin(Mathf.Deg2Rad * y_koord) * Math.Cos(Mathf.Deg2Rad
                    * latitude)) +
111             (Math.Cos(Mathf.Deg2Rad * y_koord) * Math.Cos(Mathf.Deg2Rad * stundenwinkel)
                    *
112             Math.Sin(Mathf.Deg2Rad * latitude))) / Math.Cos(Mathf.Deg2Rad * h);
```

```
113
114             double y_atan = (Math.Cos(Mathf.Deg2Rad * y_koord) * Math.Sin(Mathf.Deg2Rad *
                    stundenwinkel)) /
115             Math.Cos(Mathf.Deg2Rad * h);
116
117             double atanWert = Mathf.Rad2Deg * Math.Atan2(y_atan, x_atan) + 180;
118
119             // Umrechnung kartesische Koordinaten
120             double x_cart = Math.Round(distance * Math.Cos(Mathf.Deg2Rad * h) * Math.Cos(
                    Mathf.Deg2Rad * atanWert), 2);
121             double y_cart = Math.Round(distance * Math.Cos(Mathf.Deg2Rad * h) * Math.Sin(
                    Mathf.Deg2Rad * atanWert), 2);
122             double z_cart = Math.Round(distance * Math.Sin(Mathf.Deg2Rad * h), 2);
123
124
125             // Unterscheidung Editor oder HL2
126             #if UNITY_EDITOR
127             // Zuweisen Koordinaten Linienzug
128             constellationPoints.Add(new Vector3(Convert.ToSingle(y_cart) * faktor,
                    Convert.ToSingle(z_cart) * faktor, Convert.ToSingle(x_cart) * faktor));
129             // Herausheben der Giraffe
130             if (star.Constellation == "Cam")
131             {
132               var Giraffe = Instantiate(Sphere_Demo, new Vector3(Convert.ToSingle(y_cart)
                      , Convert.ToSingle(z_cart), Convert.ToSingle(x_cart)), Quaternion.
                      identity);
133               Giraffe.name = "___" + star.Name + "Giraffe";
134             }
135             // Teststern
136             else if(star.Name == "Deneb")
137             {
138               var Deneb = Instantiate(Sphere_Deneb, new Vector3(Convert.ToSingle(y_cart),
                       Convert.ToSingle(z_cart), Convert.ToSingle(x_cart)), Quaternion.
                      identity);
139               Deneb.name = "__" + star.Name + star.Constellation;
140             }
141             else
142             {
143               var Test = Instantiate(Sphere, new Vector3(Convert.ToSingle(y_cart),
                      Convert.ToSingle(z_cart), Convert.ToSingle(x_cart)), Quaternion.
                      identity);
144               Test.name = star.Name + star.Constellation;
145             }
146             #endif
147
148             #if ENABLE_WINMD_SUPPORT
149             // Zuweisen Koordinaten Linienzug
150             constellationPoints.Add(new Vector3(Convert.ToSingle(y_cart), Convert.
                    ToSingle(z_cart), Convert.ToSingle(x_cart)));
151             // Herausheben der Giraffe
152             if(star.Constellation == "Cam")
153             {
154               var Giraffe = Instantiate(Sphere_Demo, new Vector3(Convert.ToSingle(y_cart)
                      , Convert.ToSingle(z_cart), Convert.ToSingle(x_cart)), Quaternion.
                      identity);
155               Giraffe.name = "___" + star.Name + "Giraffe";
156             }
157             // Deneb
158             else if(star.Name == "Deneb")
159             {
160               var Deneb = Instantiate(Sphere_Deneb, new Vector3(Convert.ToSingle(y_cart),
                       Convert.ToSingle(z_cart), Convert.ToSingle(x_cart)), Quaternion.
```

```
                identity);
161             Deneb.name = "__" + star.Name + star.Constellation;
162           }
163           else
164           {
165             var Test = Instantiate(Sphere, new Vector3(Convert.ToSingle(y_cart),
                    Convert.ToSingle(z_cart), Convert.ToSingle(x_cart)), Quaternion.
                    identity);
166             Test.name = star.Name + star.Constellation;
167           }
168
169           #endif
170
171           // Kontrollstern
172           if (star.Name =="Deneb")
173           {
174             Debug.Log($"Alpha␣Cam:␣Ra␣=␣{x_koord},␣Dec␣=␣{y_koord},␣␣Azimut␣=␣{atanWert
                    },␣Hoehe␣=␣{h},␣kartesische␣Koordinaten:␣{x_cart},␣{y_cart},␣{z_cart}"
                    );
175           }
176
177           // Kontrollwerte
178           //Debug.Log($"␣Sternbild:␣{star.ID},␣{star.Constellation}␣-␣Punkte␣aus␣dem␣
                  Renderer:␣{constellationPoints[0][0]},␣" +
179           //$"x:␣{x_cart},␣y:␣{y_cart}␣\n");
180
181
182
183         }
184
185         // Linien erzeugen
186         Linienzug(constellationPoints);
187
188       }
189
190       // Kontrollwerte
191       Debug.Log($"Siderdial␣Time:␣{siderealTime}␣\n");
192       Debug.Log($"LMST:␣{lmst}␣\n");
193       Debug.Log($"Jahr:␣{year},␣Monat:␣{month},␣Tag:␣{day},␣Uhrzeit:␣{hour}:{minute}:{
              second}");
194       Debug.Log($"Laengengrad:␣{longitude}");
195
196     } // Ende void start()
197
198
199
200
201     // Methode Gitter, Quelle ChatGPT
202     void Koordinatennetz()
203     {
204       for (int lat = 0; lat <= latitudeLines; lat++)
205       {
206         float theta = lat * Mathf.PI / latitudeLines;
207         float sinTheta = Mathf.Sin(theta);
208         float cosTheta = Mathf.Cos(theta);
209
210         for (int lon = 0; lon <= longitudeLines; lon++)
211         {
212           float phi = lon * 2f * Mathf.PI / longitudeLines;
213           float sinPhi = Mathf.Sin(phi);
214           float cosPhi = Mathf.Cos(phi);
215
```

```
            float x = radius * sinTheta * cosPhi;
            float y = radius * cosTheta;
            float z = radius * sinTheta * sinPhi;

            Vector3 pointOnSphere = new Vector3(x, y, z);

            // Azimut Nord = 0
            if (Mathf.Approximately(phi, 0f) || Mathf.Approximately(phi, 2f * Mathf.PI))
            {

              Instantiate(Cylinder_0, pointOnSphere, Quaternion.identity);
            }
            else
            {

              var Streifen = Instantiate(Cylinder, pointOnSphere, Quaternion.identity);
            }
          }
        }
    }


    void Linienzug(List<Vector3> points)
    {
      LineRenderer lineRenderer = Instantiate(lineRendererPrefab, transform);
      lineRenderer.positionCount = points.Count;
      lineRenderer.SetPositions(points.ToArray());
      lineRendererPrefab.startWidth = 0.01f;
      lineRendererPrefab.endWidth = 0.01f;
    }

    List<StarData> ReadCsvFile(string filePath)
    {
      List<StarData> starDataList = new List<StarData>();

      try
      {
        var lines = File.ReadAllLines(filePath).Skip(1); // Skip header

        foreach (var line in lines)
        {
          var values = line.Split(';');

          StarData star = new StarData
          {
            ID = int.Parse(values[0]),
            Name = values[1],
            RA = double.Parse(values[2].Replace('.', ',')),
            DEC = double.Parse(values[3].Replace('.', ',')),
            Type = values[4],
            Constellation = values[5]
          };

          starDataList.Add(star);
        }
      }
      catch (Exception ex)
      {
        Debug.LogError($"Error reading CSV file: {ex.Message}");
      }

      return starDataList;
```

```
278     }
279
280
281         // Klasse Sternzeit und Julianisches Datum, Quelle zT ChatGPT
282     public static class SiderealTimeCalculator
283     {
284       public static double CalculateSiderealTime(int year, int month, int day, double
              utc, double longitude)
285       {
286         double jd = CalculateJulianDate(year, month, day);
287         double t = (jd - 2451545.0) / 36525;
288
289         // Greenwich siderial time at 0h UTC (hours)
290         double st = ((24110.54841 + 8640184.812866 * t +
291         0.093104 * t * t - 0.0000062 * t * t * t) / 3600) * 15;
292
293         double faktor = st / 360;
294         double f = faktor % 1;
295         st = f * 360;
296
297         return st;
298       }
299
300       private static double CalculateJulianDate(int year, int month, int day)
301       {
302         int y, m;
303         double jd;
304
305         if (month > 2)
306         {
307           y = year;
308           m = month;
309         }
310         else
311         {
312           y = year - 1;
313           m = month + 12;
314         }
315
316         int d = day;
317         double h = 0;
318
319         if (year <= 1582 && month <= 10 && day <= 4)
320         {
321           jd = 0;
322         }
323         else if (year == 1582 && month == 10 && day > 4 && day < 15)
324         {
325           d = 15;
326           jd = -10;
327         }
328         else
329         {
330           int a = y / 100;
331           jd = 2 - a + a / 4;
332         }
333
334         jd = 365.25 * (y + 4716) + 30.6001 * (m + 1) + d + h + jd - 1524.5;
335
336         return jd;
337       }
338     }
```

```
339
340
341       // Ende Monobehaviour
342     }
343     [System.Serializable]
344     public class StarData
345     {
346       public int ID { get; set; }
347       public string Name { get; set; }
348       public double RA { get; set; }
349       public double DEC { get; set; }
350       public string Type { get; set; }
351       public string Constellation { get; set; }
352
353     }
```

A.7. Inhalte BSC-Katalog

Listing 8: ASCII Tabelle BSC

```
Byte-by-byte Description of file: catalog
--------------------------------------------------------------------------------
Bytes Format  Units   Label    Explanations
--------------------------------------------------------------------------------
1-  4  I4     ---     HR       [1/9110]+ Harvard Revised Number
= Bright Star Number
5- 14  A10    ---     Name     Name, generally Bayer and/or Flamsteed name
15- 25  A11    ---     DM       Durchmusterung Identification (zone in
bytes 17-19)
26- 31  I6     ---     HD       [1/225300]? Henry Draper Catalog Number
32- 37  I6     ---     SAO      [1/258997]? SAO Catalog Number
38- 41  I4     ---     FK5      ? FK5 star Number
42  A1     ---     IRflag   [I] I if infrared source
43  A1     ---   r_IRflag  *[ ':] Coded reference for infrared source
44  A1     ---    Multiple *[AWDIRS] Double or multiple-star code
45- 49  A5     ---     ADS      Aitken's Double Star Catalog (ADS) designation
50- 51  A2     ---     ADScomp  ADS number components
52- 60  A9     ---     VarID    Variable star identification
61- 62  I2     h       RAh1900  ?Hours RA, equinox B1900, epoch 1900.0 (1)
63- 64  I2     min     RAm1900  ?Minutes RA, equinox B1900, epoch 1900.0 (1)
65- 68  F4.1   s       RAs1900  ?Seconds RA, equinox B1900, epoch 1900.0 (1)
69  A1     ---     DE-1900  ?Sign Dec, equinox B1900, epoch 1900.0 (1)
70- 71  I2     deg     DEd1900  ?Degrees Dec, equinox B1900, epoch 1900.0 (1)
72- 73  I2     arcmin  DEm1900  ?Minutes Dec, equinox B1900, epoch 1900.0 (1)
74- 75  I2     arcsec  DEs1900  ?Seconds Dec, equinox B1900, epoch 1900.0 (1)
76- 77  I2     h       RAh      ?Hours RA, equinox J2000, epoch 2000.0 (1)
78- 79  I2     min     RAm      ?Minutes RA, equinox J2000, epoch 2000.0 (1)
80- 83  F4.1   s       RAs      ?Seconds RA, equinox J2000, epoch 2000.0 (1)
84  A1     ---     DE-      ?Sign Dec, equinox J2000, epoch 2000.0 (1)
85- 86  I2     deg     DEd      ?Degrees Dec, equinox J2000, epoch 2000.0 (1)
87- 88  I2     arcmin  DEm      ?Minutes Dec, equinox J2000, epoch 2000.0 (1)
89- 90  I2     arcsec  DEs      ?Seconds Dec, equinox J2000, epoch 2000.0 (1)
91- 96  F6.2   deg     GLON     ?Galactic longitude (1)
97-102  F6.2   deg     GLAT     ?Galactic latitude (1)
103-107  F5.2   mag     Vmag     ?Visual magnitude (1)
108  A1     ---   n_Vmag    *[ HR] Visual magnitude code
109  A1     ---   u_Vmag     [ :?] Uncertainty flag on V
110-114  F5.2   mag     B-V      ? B-V color in the UBV system
115  A1     ---   u_B-V      [ :?] Uncertainty flag on B-V
116-120  F5.2   mag     U-B      ? U-B color in the UBV system
121  A1     ---   u_U-B      [ :?] Uncertainty flag on U-B
122-126  F5.2   mag     R-I      ? R-I   in system specified by n_R-I
127  A1     ---   n_R-I      [CE:?D] Code for R-I system (Cousin, Eggen)
128-147  A20    ---     SpType   Spectral type
148  A1     ---   n_SpType   [evt] Spectral type code
149-154  F6.3 arcsec/yr pmRA    *?Annual proper motion in RA J2000, FK5 system
155-160  F6.3 arcsec/yr pmDE     ?Annual proper motion in Dec J2000, FK5 system
161  A1     ---   n_Parallax [D] D indicates a dynamical parallax,
otherwise a trigonometric parallax
162-166  F5.3   arcsec  Parallax ? Trigonometric parallax (unless n_Parallax)
167-170  I4     km/s    RadVel   ? Heliocentric Radial Velocity
171-174  A4     ---   n_RadVel  *[V?SB123O ] Radial velocity comments
175-176  A2     ---   l_RotVel   [<=> ] Rotational velocity limit characters
177-179  I3     km/s    RotVel   ? Rotational velocity, v sin i
180  A1     ---   u_RotVel   [ :v] uncertainty and variability flag on
RotVel
181-184  F4.1   mag     Dmag     ? Magnitude difference of double,
or brightest multiple
```

```
185-190  F6.1   arcsec  Sep      ? Separation of components in Dmag
if occultation binary.
191-194  A4     ---     MultID   Identifications of components in Dmag
195-196  I2     ---     MultCnt  ? Number of components assigned to a multiple
197  A1     ---     NoteFlag [*] a star indicates that there is a note
(see file notes)
```

A.8. Liste der Sternbilder

[ht]

Constellation	Abbreviations IAU	Origin
Andromeda	And	ancient (Ptolemy)
Antlia	Ant	1763, Lacaille
Apus	Aps	1603, Uranometria, created by Keyser and de Houtman
Aquarius	Aqr	ancient (Ptolemy)
Aquila	Aql	ancient (Ptolemy)
Ara	Ara	ancient (Ptolemy)
Aries	Ari	ancient (Ptolemy)
Auriga	Aur	ancient (Ptolemy)
Boötes	Boo	ancient (Ptolemy)
Caelum	Cae	1763, Lacaille
Camelopardalis	Cam	1613, Plancius
Cancer	Cnc	ancient (Ptolemy)
Canes Venatici	CVn	1690, Firmamentum Sobiescianum, Hevelius
Canis Major	CMa	ancient (Ptolemy)
Canis Minor	CMi	ancient (Ptolemy)
Capricornus	Cap	ancient (Ptolemy)
Carina	Car	1763, Lacaille, split from Argo Navis
Cassiopeia	Cas	ancient (Ptolemy)
Centaurus	Cen	ancient (Ptolemy)
Cepheus	Cep	ancient (Ptolemy)
Cetus	Cet	ancient (Ptolemy)
Chamaeleon	Cha	1603, Uranometria, created by Keyser and de Houtman
Circinus	Cir	1763, Lacaille
Columba	Col	1592, Plancius, split from Canis Major
Coma Berenices	Com	1536, Caspar Vopel, split from Leo
Corona Australis	CrA	ancient (Ptolemy)
Corona Borealis	CrB	ancient (Ptolemy)

Constellation	Abbreviations IAU	Origin
Corvus	Crv	ancient (Ptolemy)
Crater	Crt	ancient (Ptolemy)
Crux	Cru	1603, Uranometria, split from Centaurus
Cygnus	Cyg	ancient (Ptolemy)
Delphinus	Del	ancient (Ptolemy)
Dorado	Dor	1603, Uranometria, created by Keyser and de Houtman
Draco	Dra	ancient (Ptolemy)
Equuleus	Equ	ancient (Ptolemy)
Eridanus	Eri	ancient (Ptolemy)
Fornax	For	1763, Lacaille
Gemini	Gem	ancient (Ptolemy)
Grus	Gru	1603, Uranometria, created by Keyser and de Houtman
Hercules	Her	ancient (Ptolemy)
Horologium	Hor	1763, Lacaille
Hydra	Hya	ancient (Ptolemy)
Hydrus	Hyi	1603, Uranometria, created by Keyser and de Houtman
Indus	Ind	1603, Uranometria, created by Keyser and de Houtman
Lacerta	Lac	1690, Firmamentum Sobiescianum, Hevelius
Leo	Leo	ancient (Ptolemy)
Leo Minor	LMi	1690, Firmamentum Sobiescianum, Hevelius
Lepus	Lep	ancient (Ptolemy)
Libra	Lib	ancient (Ptolemy)
Lupus	Lup	ancient (Ptolemy)
Lynx	Lyn	1690, Firmamentum Sobiescianum, Hevelius
Lyra	Lyr	ancient (Ptolemy)
Mensa	Men	1763, Lacaille, as Mons Mensæ
Microscopium	Mic	1763, Lacaille
Monoceros	Mon	1613, Plancius

Constellation	Abbreviations IAU	Origin
Musca	Mus	1603, Uranometria, created by Keyser and de Houtman
Norma	Nor	1763, Lacaille
Octans	Oct	1763, Lacaille
Ophiuchus	Oph	ancient (Ptolemy)
Orion	Ori	ancient (Ptolemy)
Pavo	Pav	1603, Uranometria, created by Keyser and de Houtman
Pegasus	Peg	ancient (Ptolemy)
Perseus	Per	ancient (Ptolemy)
Phoenix	Phe	1603, Uranometria, created by Keyser and de Houtman
Pictor	Pic	1763, Lacaille, as Equuleus Pictoris
Pisces	Psc	ancient (Ptolemy)
Piscis Austrinus	PsA	ancient (Ptolemy)
Puppis	Pup	1763, Lacaille, split from Argo Navis
Pyxis	Pyx	1763, Lacaille
Reticulum	Ret	1763, Lacaille
Sagitta	Sge	ancient (Ptolemy)
Sagittarius	Sgr	ancient (Ptolemy)
Scorpius	Sco	ancient (Ptolemy)
Sculptor	Scl	1763, Lacaille
Scutum	Sct	1690, Firmamentum Sobiescianum, Hevelius
Serpens	Ser	ancient (Ptolemy)
Sextans	Sex	1690, Firmamentum Sobiescianum, Hevelius
Taurus	Tau	ancient (Ptolemy)
Telescopium	Tel	1763, Lacaille
Triangulum	Tri	ancient (Ptolemy)
Triangulum Australe	TrA	1603, Uranometria, created by Keyser and de Houtman
Tucana	Tuc	1603, Uranometria, created by Keyser and de Houtman
Ursa Major	UMa	ancient (Ptolemy)

Constellation	Abbreviations IAU	Origin
Ursa Minor	UMi	ancient (Ptolemy)
Vela	Vel	1763, Lacaille, split from Argo Navis
Virgo	Vir	ancient (Ptolemy)
Volans	Vol	1603, Uranometria, created by Keyser and de Houtman, as Piscis Volans
Vulpecula	Vul	1690, Firmamentum Sobiescianum, Hevelius, as Vulpecula cum Ansere

Tabelle 19: Die offizielle Liste der Sternbilder. Der lateinische Name in der ersten Spalte, die entsprechende Abkürzung in Spalte 2. In der dritten Spalte ist der Ursprung des Sternbilds skizziert.

Literaturverzeichnis

Bahri, H., Krcmarik, D., Moezzi, R., and Kočí, J. (2019). Efficient use of mixed reality for bim system using microsoft hololens. *IFAC-PapersOnLine*, 52(27):235–239. 16th IFAC Conference on Programmable Devices and Embedded Systems PDES 2019.

Bertold Witte, P. S. (2015). *Vermessungskunde und Grundlagen der Statistik für das Bauwesen*. Wichmann.

Bitschi, D., Fürmetz, J., Gilbert, F., Jörgens, M., Watrinet, J., Pätzold, R., Lang, C., Neidlein, C., Böcker, W., and Bormann, M. (2023). Preoperative mixed-reality visualization of complex tibial plateau fractures and its benefit compared to ct and 3d printing. *Journal of Clinical Medicine*, 12(5).

Boyce, M. W., Thomson, R. H., Cartwright, J. K., Feltner, D. T., Stainrod, C. R., Flynn, J., Ackermann, C., Emezie, J., Amburn, C. R., and Rovira, E. (2022). Enhancing military training using extended reality: A study of military tactics comprehension. *Frontiers in Virtual Reality*, 3.

Emerson, B. (1973). A method of calculating apparent places of stars. *Royal Greenwich Observatory Bulletins*, 178:299–300.

finance.yahoo.com (2023). Microsoft's headache-inducing army goggles delayed for at least two years. https://finance.yahoo.com/news/microsoft-headache-inducing-army-goggles-205417485.html.

golem.de (2023). Microsofts kopfschmerz-kampfbrille um zwei jahre verschoben. https://www.golem.de/news/ivas-microsofts-kopfschmerz-kampfbrille-um-zwei-jahre-verschoben-2304-173826.html.

Gullberg, S. R., Hamacher, D. W., Martín-Lopez, A., Mejuto, J., Munro, A. M., and Orchiston, W. (2020). A cultural comparison of the 'dark constellations' in the Milky Way. *Journal of Astronomical History and Heritage*, 23(2):390–404.

Guonian Lü, Michael Batty, J. S. H. L. A.-X. Z. and Chen, M. (2019). Reflections and speculations on the progress in geographic information systems (gis): a geographic perspective. *International Journal of Geographical Information Science*, 33(2):346–367.

Hammady, R., Ma, M., and Strathearn, C. (2019). User experience design for mixed reality: a case study of hololens in museum. *International Journal of Technology Marketing*, 13(3-4):354–375.

Hanslmeier, A. (2020). *Einführung in Astronomie und Astrophysik*. Springer Berlin Heidelberg;Springer Spektrum, 4. aufl. edition.

Hilton, J. L., Capitaine, N., Chapront, J., Ferrandiz, J. M., Fienga, A., Fukushima, T., Getino, J., Mathews, P., Simon, J. L., Soffel, M., Vondrak, J., Wallace, P., and Williams, J. (2006). Report of the international astronomical union division i working group on precession and the ecliptic. *Celestial Mechanics and Dynamical Astronomy*, 94(3):351–367.

Hoffleit, D. and Jaschek, C. (1991). *The Bright star catalogue*.

Hoffmann, S. M. (2021). *Wie der Löwe in den Himmel kam*. Kosmos.

IAU (2023). The constellations (iau). https://www.iau.org/public/themes/constellations/.

Jenkins, M., Wollocko, A., Negri, A., and Ficthl, T. (2018). Augmented reality and mixed reality prototypes for enhanced mission command/battle management command and control (bmc2) execution. In Chen, J. Y. and Fragomeni, G., editors, *Virtual, Augmented and Mixed Reality: Applications in Health, Cultural Heritage, and Industry*, pages 272–288, Cham. Springer International Publishing.

Karttunen, H. (2005). *Fundametal Astronomy*. Springer Verlag.

Kunnen, S., Adamenko, D., Pluhnau, R., Loibl, A., and Nagarajah, A. (2020). System-based concept for a mixed reality supported maintenance phase of an industrial plant. *Procedia CIRP*, 91:15–20. Enhancing design through the 4th Industrial Revolution Thinking.

Künzl, E. (2005). *Himmelsgloben und Sternkarten*. Theiss.

Leger, E., Drouin, S., Collins, D. L., Popa, T., and Kersten-Oertel, M. (2017). Quantifying attention shifts in augmented reality image-guided neurosurgery. *Healthcare Technology Letters*, 4(5):188–192.

Lieske, J. H., Lederle, T., Fricke, W., and Morando, B. (1977). Expressions for the precession quantities based upon the iau (1976) system of astronomical constants. *aap*.

Maniam, P., Schnell, P., Dan, L., Portelli, R., Erolin, C., Mountain, R., and Wilkinson, T. (2020). Exploration of temporal bone anatomy using mixed reality (hololens): development of a mixed reality anatomy teaching resource prototype. *Journal of Visual Communication in Medicine*, 43(1):17–26. PMID: 31645155.

Meeus, J. (1998). *Astronomical Algorithms*. William Bell, Inc.

Microsoft (2023a). Coordinate systems. https://learn.microsoft.com/en-us/windows/mixed-reality/design/coordinate-systems.

Microsoft (2023b). Microsoft hololens. https://news.microsoft.com/de-de/microsoft-erklaert-was-ist-microsoft-hololens-definition-funktionen/.

Microsoft (2023c). Willkommen beim mixed reality-featuretool. https://learn.microsoft.com/de-de/windows/mixed-reality/develop/unity/welcome-to-mr-feature-tool.

Milgram, P., Takemura, H., Utsumi, A., and Kishino, F. (1994). Augmented reality: A class of displays on the reality-virtuality continuum. *Telemanipulator and Telepresence Technologies*, 2351.

Montenbruck, O. (2005). *Grundlagen der Ephemeridenrechnung*. Springer Verlag.

Navratil, G., Konturek, P., and Giannopoulos, I. (2020). Interacting with 3d models – 3d-cad vs. holographic models. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, VI-4/W1-2020:129–134.

NOAJ (2023). Ephemeris computation office. https://eco.mtk.nao.ac.jp/koyomi/index.html.en.

Park, S., Bokijonov, S., and Choi, Y. (2021). Review of microsoft hololens applications over the past five years. *Applied Sciences*, 11(16).

Peterson, B. N., Hitching, R., Howard, L., Zhu, K., Fontenot, M. R., Alhalabi, W., Seibel, A., Harris, O. A., Madrigal, E., Adamson, M. M., and Hoffman, H. G. (2021). Immersive virtual reality: A safe, scalable, non-opioid analgesic for military and veteran patients. *Frontiers in Virtual Reality*, 2.

Richards, S. (2020). Student engagement using hololens mixed-reality technology in human anatomy laboratories for osteopathic medical students: an instructional model.

Richter, D. (2005). *Ephemeridenrechnung Schritt für Schritt*. Springer Verlag.

Saastamoinen, J. (1972). Contributions to the theory of atmospheric refraction.

Schilling, G. (2019). *Constellations - A Story of Space told through the 88 known star patterns in the night sky*. Black Dog and Leventhal.

Silva, H., Resende, R., and Breternitz, M. (2018). Mixed reality application to support infrastructure maintenance. In *2018 International Young Engineers Forum (YEF-ECE)*, pages 50–54.

Snyder, J. P. (1987). *Map projections: A working manual*. U.S. Government Printing Office.

Sutherland, I. E. (1968). A head-mounted three dimensional display. In *Proceedings of the December 9-11, 1968, Fall Joint Computer Conference, Part I*, AFIPS '68 (Fall, part I), page 757–764, New York, NY, USA. Association for Computing Machinery.

Technologies, U. (2024). Streaming assets. https://docs.unity3d.com/Manual/StreamingAssets.html.

Ullo, S. L., Piedimonte, P., Leccese, F., and De Francesco, E. (2019). A step toward the standardization of maintenance and training services in c4i military systems with mixed reality application. *Measurement*, 138:149–156.

Unsöld, B. (2005). *Der neue Kosmos*. Springer.

Urban, S. and Seidelmann, P. (2012). *Explanatory Supplement to the Astronomical Almanac*. University Science Books.

Vogtherr, T. (2012). *Zeitrechnung - Von den Sumerern bis zur Swatch*. C.H. Beck Wissen.

Wang, G., Qian, Y., Zhang, Y., Lu, Z., Chen, X., and Liu, D. (2019). Design and implementation of children's games based on mixed reality. In *2019 IEEE International Conference on Computer Science and Educational Informatization (CSEI)*, pages 176–180.

Wang, W., Wu, X., Chen, G., and Chen, Z. (2018). Holo3dgis: Leveraging microsoft hololens in 3d geographic information. *ISPRS International Journal of Geo-Information*, 7(2).

Xue, H., Sharma, P., and Wild, F. (2019). User satisfaction in augmented reality-based training using microsoft hololens. *Computers*, 8(1).

Zhang, L., Chen, S., Dong, H., and El Saddik, A. (2018). Visualizing toronto city data with hololens: Using augmented reality for a city model. *IEEE Consumer Electronics Magazine*, 7(3):73–80.

Çöltekin, A., Lochhead, I., Madden, M., Christophe, S., Devaux, A., Pettit, C., Lock, O., Shukla, S., Herman, L., Stachoň, Z., Kubíček, P., Snopková, D., Bernardes, S., and Hedley, N. (2020). Extended reality in spatial sciences: A review of research challenges and future directions. *ISPRS International Journal of Geo-Information*, 9(7).