Tampere University

Christian Kaarre

HARDWARE FOR MOBILE POSITIONING

Considerations on Design and Development of Cost-Efficient Solutions

Master of Science Thesis
Faculty of Information Technology and Communication Sciences
Examiners: Prof. Esa Rahtu
Prof. Joni–Kristian Kämäräinen
September 2022

ABSTRACT

Christian Kaarre: Hardware for Mobile Positioning
Master of Science Thesis
Tampere University
September 2022

The estimation of a moving agent's position in an unknown environment is a problem formulated in its current form already in the 1980s. Emphasis on localization and mapping problems has grown rapidly in the last two decades driven by the increased computational capability of especially handheld systems and a large number of target applications in various fields, ranging from self-driving cars and geomatics to robotics and virtual/augmented reality. Besides the algorithms for positioning, hardware plays a major role as a backbone for enabling accurate, robust and flexible position estimation solutions.

This thesis gives an overview of sensors utilized in mobile positioning with a focus on passive visual-inertial sensors as an alternative to more expensive active-ranging solutions. The main research interest of the thesis is the feasibility of developing and implementing a cost-efficient hardware solution for positioning. Visual, inertial and satellite positioning sensors' advantages, performance parameters, sources of error and physical requirements are considered. Sensor integration and both sensor and system-level calibration in a multisensor setup are discussed. Levels of developer involvement and options for hardware development approaches are presented, mainly ready-made modular solutions, building on top of intermediate products and development from scratch.

Hardware development processes are demonstrated by implementing a synchronized visual-inertial positioning system including two pairs of stereo cameras, an inertial measurement unit and a Real-Time Kinematic capable satellite positioning solution. The system acts as a cost-efficient example for options and decisions required on the selection of sensors and computational subsystems supporting the sensor hardware, integration and continuous temporal synchronization of sensors as well as requirements and manufacturing options for system enclosures. Even though the direct costs of the solution seem inexpensive compared to competitive solutions, accounting for the development time and associated risk makes hardware development from scratch less attractive option compared to other approaches. For a proof-of-concept or a case in which a very limited number of end products are produced, implementation from the ground up is most likely time-consuming and thus ends up being an expensive endeavor compared to other approaches. Also, the benefits of control over the details of hardware and integration may not be fully utilized.

Keywords: positioning, hardware, visual–inertial, camera, VIO, RTK, GNSS, IMU

TIIVISTELMÄ

Christian Kaarre: Liikkuvan Paikannuksen Laitteisto
Diplomityö
Tampereen yliopisto
Syyskuu 2022

Liikkuvan agentin sijainnin estimointi tuntemattomassa ympäristössä on jo 1980-luvulla nykyisessä muodossaan esitetty ongelma. Erityisesti kannettavien laitteiden laskentatehon kasvaminen sekä sovelluskohteiden laaja kirjo, itseajavista autoista ja maanmittauksesta robotiikkaan ja virtuaaliseen sekä lisättyyn todellisuuteen, painotus paikannukseen ja kartoittamiseen (engl. SLAM) liittyviin ongelmiin on viime vuosikymmeninä kasvanut merkittävästi. Paikannuksessa käytettävien algoritmien lisäksi laitteistolla on suuri rooli entistä tarkempien, virhetilanteita kestävämpien ja käyttökohteiltaan joustavampien paikannusratkaisujen runkona.

Opinnäytteessä käydään läpi liikkuvassa paikannuksessa käytettäviä sensorikokonaisuuksia keskittyen passiivisiin visuaali–inertiaalijärjestelmiin kustannustehokkaana vaihtoehtona aktiivisia syvyyssensoreita hyödyntäviin ratkaisuihin. Tutkimuksen kannalta tärkein päämäärä on kartoittaa kustannustehokkaan paikannusjärjestelmän toteuttamiskelpoisuutta ja toteutuksen vaatimuksia. Visuaalisten, inertiaalisten sekä satelliittipaikannussensorien hyötyjä, suorituskykyparametrejä, virhelähteitä ja fyysisiä ominaisuuksia tarkastellaan. Työssä käsitellään myös sensorien integraatiota ja sekä sensori että järjestelmätasoista kalibrointia. Laitteiston kehittämisen vaativuutta ja vaihtoehtoja lähestymistavoille laitteistokehitykseen pohditaan. Laitteistokehityksessä käytettävät lähestymistavat jaotellaan työssä valmiisiin modulaarisiin ratkaisuihin, puolivalmisteiden päälle rakennettaviin ratkaisuihin sekä laitteiston kehittämiseen tyhjästä ilman valmiita osakokonaisuuksia.

Laitteiston kehittämisprosesseja demonstroidaan opinnäytteessä suunnittelemalla ja toteuttamalla synkronoitu visuaali–inertiaalisensoreita käyttävä paikannusjärjestelmä, sisältäen kaksi paria stereokameroita, inertiaalimittausyksikön sekä reaaliaikaiseen kinemaattiseen mittaukseen kykenevän satelliittipaikannusratkaisun. Toteutettua järjestelmää voidaan käsitellä kustannustehokkaana esimerkkinä laitteiston toteuttamisen vaihtoehdoista ja päätöksenteon osatekijöistä koskien sensorien ja laskentayksiköiden valintaa, integraatiota ja sensorien ajallista synkronoisointia sekä laitteiston kotelointiin ja valmistukseen liittyviä vaihtoehtoja ja vaatimuksia. Vaikka toteutuksen suorat laitteistokustannukset saavat laitteiston osakokonaisuudet näyttämään edullisilta verrattuna valmiisiin kilpailukykyisiin ratkaisuihin, laitteiston tyhjästä rakentamisen houkuttelevuus laskee merkittävästi pohdittaessa kehitykseen liittyviä epäsuoria kustannuksia ja riskejä. Laitteistoratkaisun toteuttaminen tyhjästä on todennäköisesti aikaakuluttava ja näin ollen kallis vaihtoehto verrattuna muihin tarkasteltuihin lähestymistapoihin mikäli laitteisto toteutetaan konseptia varten tai lopputuotteiden tuotantomäärä on erittäin rajattu. Myöskään tyhjästä rakentamisen hyötyjä laitteistokontrollin ja integraation näkökulmista ei välttämättä ole mahdollista hyödyntää täysin.

Avainsanat: paikannus, laitteisto, visuaali–inertiaali, kamera, VIO, RTK, GNSS, IMU

PREFACE

I would like to thank my supervisor professor Esa Rahtu for his guidance, support and patience with the research project and this thesis. I would also like to thank people from SpectacularAI and Aalto University's Machine Learning and Computer Vision groups, especially Dr. Seiskari and professors Solin and Kannala, for their valuable contributions and feedback to the project.

Special thanks to Centre for Immersive Visual Technologies (CIVIT) at Tampere University and Leica Geosystems Oy for lending equipment and providing helpful insight on the hardware as well as FabLab Tampere's staff for their help with prototype manufacturing. I want to thank professor Joni Kämäräinen for examining the finished thesis and everyone at Tampere University who I've had the pleasure of working with during my studies.

Tampere, 30th September 2022

Christian Kaarre

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

ADC	Analog-to-Digital Converter
ADR	Automotive Dead–Reckoning
AR	Augmented Reality
ASIC	Application Specific Integrated Circuits
BMS	Battery Management Unit
BoM	Bill-of-Materials
C-PHY	Camera Physical Layer
C/A	Coarse/Acquisition
CAD	Computer Aided Design
CCD	Charge-Coupled Device
CMOS	Complementary Metal-Oxide Semiconductor
CSI	Camera Serial Interface
D-PHY	Display Physical Layer
DDS	Data Distribution Service
DGNSS	Differential GNSS
DOP	Dilution of Perception
DSLR	Digital Single-Lens Reflex
DSP	Digital Signal Processor
ECC	Error Correction Code
EMVA	European Machine Vision Association
FFC	Flat Flex Cable
FPU	Floating-Point Unit
FTDI	Future Technology Devices International Limited
GenICam	GENeric Interface for CAMeras
GenTL	GENeric Transport Layer
GNSS	Global Navigation Satellite System
GPIO	General Purpose Input Output

GPS	Global Positioning System
I2C	Inter-Integrated Circuit
IMU	Inertial Measurement Unit
LIDAR	Light Detection and Ranging
LIO	LIDAR-Inertial Odometry
LLP	Low-Level Protocol
MAC	Media Access Control address
MCU	MicroController Unit
MEMS	Micro-Electro Mechanical Systems
MIPI	Mobile Industry Processor Interface
MPU	MicroProcessing Unit
NTRIP	Network Transport of RTCM via Internet Protocol
NVMe	NonVolatile Memory Express
PCB	Printed Circuit Board
PCIe	Peripheral Component Interconnect Express
PoE	Power Over Ethernet
PPP	Precise Point Positioning
PRN	Pseudorandom Noise
ROS	Robotic Operating System
RTCM	Radio Technical Commission for Maritime Services
RTK	Real Time Kinematic
RTOS	Real-Time Operating System
SBAS	Satellite Based Augmentation System
SFNC	Standard Features Naming Convention
SLAM	Simultaneous Localization and Mapping
SNR	Signal-to-Noise Ratio
SPI	Serial Peripheral Interface
SSD	Solid-State Drive
TCP/IP	Transmission Control Protocol / Internet Protocol
ToF	Time of Flight
TTL	Transistor-Transistor Logic
UART	Universal Asynchronous Receiver-Transmitter

UDP	User Datagram Protocol
USB	Universal Serial Bus
UTC	Coordinated Universal Time
VIO	Visual-Inertial Odometry
VR	Virtual Reality
VRS	Virtual Reference Station

1. INTRODUCTION

Recent advances made on methods for localization driven by a need for accurate, robust and cost-effective solutions are apparent [1]. Applications for cheaper positioning solutions include autonomous [2, 3, 4] and unmanned vehicles [5], robotics and augmented/virtual reality (AR/VR) in consumer and industrial segments [6, 7]. While methods and algorithms for such applications are quite a popular area of research in academia, hardware solutions for positioning may often be overlooked. A way of combining hardware and academic research on positioning solutions is by published datasets, yet a common approach taken in composing the dataset's hardware is utilizing expensive enterprise-level integrated solutions [8]. More cost-efficient alternatives for individual sensors exist, the drawback of more affordable solutions being a lack of expandability and possibilities of integration with other sensors. These approaches leave a gap in the implementation of positioning solutions concerning the processes and options for cost-efficient and expandable systems.

The target of this thesis is to explore the possibilities of combining individual sensors into a synchronized solution. The main research question is is it feasible to design and implement a cost-efficient solution for positioning applications? If so, what are the requirements for a competitive solution in terms of sensor selection, integration and performance? Can the development processes and system design be standardized?

Working principles and performance parameters of visual, inertial and satellite positioning sensors are presented and sensor synchronization and calibration are discussed in the chapter 2. In the chapter 3 different approaches for hardware development are considered and a solution for cost-efficient hardware implementation is presented. Chapter 4 goes through the implementation of a visual-inertial odometry system and different options for the implementation are discussed. The final chapter includes conclusions and future work.

2. BACKGROUND

The chapter gives an overview of working principles and notable performance parameters of vision, inertial and satellite-based positioning sensors. Concepts regarding synchronization and calibration in a multi-sensor system are presented.

2.1 Poistioning Fundamentals

Mobile localization aims to determine the location of a moving agent with sensors [9]. A general term for sensor-based localization methods is odometry. A simple odometry model includes the agent's previous location and measurement from an arbitrary sensor measuring a change from the previous location to the current state of the agent, the current location being the previous state with added measurement-based change in location. Information extracted from the environment, known landmarks and/or measured changes in the state of the agent used in determining a location estimate is inherently imperfect introducing incrementally increasing uncertainty to the estimate [10, 11]. Another source of uncertainty besides inaccuracies present in perceiving and quantifying an environment with sensors are imperfect models for modeling the agent's movement.

Simultaneous Localization and Mapping (SLAM) targets simultaneously both odometry and real-time mapping problems. Generating a map of the environment from the sensor measurements and actively detecting previously visited locations can decrease the uncertainty of an agent's position and already travelled trajectory in case of detecting a *loop closure*, i.e. previously visited known location [12]. Uncertainty can also be decreased by increasing the quality of sensor data or fusing together multiple data sources with complementary properties, a practice generally referred to as *sensor fusion* [9, 13]. As an example of an advantage of using complementary sensors, satellite positioning can provide an accurate absolute large-scale position outdoors, while an inertial sensor's small-scale accuracy and responsiveness cannot be achieved with satellite positioning alone [14].

Widely adapted categories of sensors for mobile positioning include depth, vision and inertial as well as satellite-based positioning. Light Detection and Ranging (LIDAR) sensors fused with satellite positioning and inertia information have been successfully applied as an accurate solution for outdoors positioning due to robustness against varying conditions [15]. Another widely applied approach is the utilization of passive vision sensors in place of the active ranging sensor [16] for gaining knowledge of the environment. Information about the environment on the sensor level is less descriptive and thus relies on extracting information from sensor data for pose estimation. Differences in deploying LIDAR-Inertial (LIO) and Visual-Inertial Odometry (VIO) systems come down to the target application and the cost of the sensors. Lowering the cost of positioning solutions relies in part on the utilization of more affordable sets of sensors. VIO methods have proven to be a cost-effective alternative to LIDAR-based solutions indoors and major improvements have been made in outdoors positioning performance [17, 18, 19, 20].

The following notations will be used in this thesis. An agent's location can be represented with or without heading information. A three-dimensional estimate is referred to as *position* with x, y, z components and a six-dimensional estimate as *pose* including agent's orientation with position. Degrees of Freedom (DoF) is a general term for defining the complexity of an object's possible motion, increasing in measurable dimensions by which a system can move or the system's movement can be measured with more degrees of freedom. Similarly to position and pose, degrees of freedom could include and be represented in translational (3DoF) or both translational and rotational components (6DoF).

2.2 Vision Sensors

Visual, vision or image sensors include a variety of sensors with the ability to capture and transform light impulses into quantifiable entities. Visual sensors can be divided into area and line scan cameras depending on the output of the sensor, with two- and one-dimensional outputs respectively. Special categories of area scan cameras include neuromorphic cameras or eventcameras [21], and thermographic cameras [22] detecting infrared wavelengths. Neuromorphic cameras have been proven effective in positioning [23] while thermographic cameras are useful in applications mostly other than directly positioning, for example in pedestrian detection, search-and-rescue and surveillance. The main interest of this thesis is general-purpose area scan cameras.

2.2.1 Image Sensor

Section refers to chapters four and five of *Image Sensors and Signal Processing for Digital Still Cameras* [24] by Nakamura. The operational requirements for a camera are capturing a set of focused light rays from a scene by exposing the sensor to the light for a controlled time, converting the captured light from charges to digital quantities and transferring the digital information forward. An image sensor is responsible for the capturing process while camera optics, including a lens or a system of lenses and smaller pixel-level microlenses, focus rays of light from a scene into the sensor.

Two different sensor technologies include more traditional and already mostly abandoned charge-coupled device (CCD) and complementary metal-oxide semiconductor (CMOS). With CCD sensors charges from the pixels are transferred to the amplifier and analog-to-digital converter (ADC) via neighboring pixels and thus the amplification and the analog-to-digital conversion will be executed away from the individual photosites. With CMOS sensors the transistors for charge amplification are placed on photosites and charges are transferred to ADC per pixel basis after initial amplification. After ADC a digital signal processor (DSP) could be integrated with the sensor for managing the output, for example debayering process with color sensors.

Compared to CMOS sensors, CCD requires more power and the cost of the sensor is higher due to a less efficient and highly specialized manufacturing process. The manufacturing process of CMOS image sensors resembles the manufacturing processes of microprocessors making integration of circuitry into a sensor chip easy compared to CCD's differentiation of photosites and other circuitry. The advantages of CCD compared to earlier CMOS sensors were higher Signal-to-Noise Ratio (SNR), sensitivity and dynamic range due to external peripherals and higher fill factor. Cheaper manufacturing processes and advances in CMOS technology [25], for example backside illuminated sensors increasing fill factor [26, ch 4], have resulted in current CMOS sensors exceeding the initial advantages of CCD sensors in most domains [27].

2.2.2 Sensor Performance

Section refers to chapter three of *Image Sensors and Signal Processing for Digital Still Cameras* [24] by Nakamura. The main goal for imaging is an accurate representation of the scene without aberrations or artifacts from the sensor or optics. Performance parameters of imaging sensors can be fixed properties of the on-dye structure's implementation with manufacturer-specified properties or user-defined image processing parameters.

Starting with the exposure of a sensor, the function of a shutter is the exposure of the image sensor for a fixed time. Mechanical shutters used in Digital Single-Lens Reflex (DSLRs) cameras demonstrate the mechanical shutter's action of exposing the image sensor for a specific exposure time by physically opening and closing two sets of shutter curtains when taking a picture. As the rate of capture increases, shutter curtains act more and more in a line scan fashion. An electronic shutter is a way of starting and ending exposure without any moving parts by allowing voltage to pass through the photodiode when the image is not being captured. The main difference in shutters for capturing images without artifacts and as close to being exposed in the same timeframes comes down to how the exposure and reading of charges are executed on the sensor level. Electronic shutters in general purpose cameras fall into rolling and global shutters. A rolling shutter will activate photodiodes starting on the first row and advancing to the next rows until the whole sensor has been read keeping the exposure time for each row fixed. The main benefit of this approach is the continuous data stream coming from the sensor resulting in less buffering of the charges required on pixel-level. The global shutter on the other hand will start the exposure of the whole matrix of pixels at once and end the exposure at the same time. The downsides of temporal coherence are opposite to the rolling shutter as the sensor will be read after the exposure is stopped at the same time for each photodiode. The transfer speed of the charges to the analog-to-digital converter (ADC) is limited and thus memory will have to be implemented on individual photosites.

Physical parameters include sensor format with resolution leading to pixel size and pitch. Sensor format notates the size and ratio of the sensor, determined by the width, height and pixel size. Implemented stack of on-dye circuitry including transistors for amplification, memory and functionality for setting and resetting individual pixels for shuttering. Based on the physical capabilities and structure of a sensor notable parameters on the functional level are dynamic range, quantum efficiency (QE), bit-depth, sensitivity and signal-to-noise ratio (SNR). Dynamic range means the sensor's capturing ability in the ratio of brightest and dimmest points in a scene. Quantum efficiency represents how many photons are converted to electrons in the photodiode. Bit-depth is the quantization precision when converting an analog signal to digital, i.e. accuracy of reproducing the number of charges captured with the photodiode in binary format. Sensitivity is the lowest amount of light-producing charge over a noise wall in the photodiode. Noise is present in various types, e.g. photon noise, read noise and dark noise, and operating conditions change how the noise appears on the images, for example in temporal, spatial or frequency domains.

Color filters in front of a sensor will differentiate the wavelengths of incoming light producing color images. An example of a color filter design is the Bayer pattern, in which typically red, green and blue (RGB) wavelengths are captured. Other arrangements and variations between manufacturers exist in filter layout and ranges of filtered wavelength. The practical implication is that the pixel size for equal color and monochromatic sensors needs to be larger or the exposure time longer to capture the same amount of light.

The maximum framerate of a sensor is limited by sensor structure, mainly circuitry's ability to convert, store and transfer charges and the rate of reading the images from the sensor. User-defined artificial processing parameters include gain, brightness and contrast. These user-defined parameters affect a target application's overall performance [28].

2.2.3 Optics Performance

Section refers to the second chapter of *Image Sensors and Signal Processing for Digital Still Cameras* [24] by Nakamura. In imaging, optics refers typically to a single optical element as a lens or a collection of optical elements or a system of lenses as an objective.

An ideal optics system should render points as points, planes as planes and a subject and its image are the same shape. Since no lens or objective is ideal in a real-world scenario, the distortions and aberrations need to be modeled.

Optical aberrations are a result of imperfections of the optical elements and physical restrictions on the refraction of light rays through a lens. Outcomes of aberrations are inconsistent focus on the image plane, known as spherical, coma, astigmatism and field curvature aberrations, and varying refraction of different wavelengths of light based on the angle of light i.e. chromatic aberration. Even though an ideal optical system is free of any distortions, distortions could be seen as a feature of optics. In specific applications when a wide Field-of-View (FoV) is required, distortions cannot be fully eliminated. Two main categories of distortions are radial and tangential. Radial distortion range from positive, i.e. pincushion, to negative, i.e. barrel, distortion with combinations of both distortions on a single optical element existing. Radial distortions bend lines in an image, while tangential distortion is present when the image plane is not aligned with the lens, i.e. an orthogonal line from the image plane is non-parallel to the principal axis of a lens.

The two main lens parameters are aperture and focal length. Aperture controls the cone of incoming light rays thus by increasing the opening more light is entering the image plane through the lens. Aperture is typically expressed in F-numbers where a higher F-number means a smaller aperture. Focal length is a measurement from an image plane to a lens' principal plane and defines the focus distance, magnification and field-of-view. Both aperture and focal length affect the depth-of-field of a camera. The calibration of intrinsic camera parameters aims to determine the distortions of an optical system and optics in relation to a physical image plane. Simultaneously the focal length, the optical center of the image plane and pixel skew are determined [29, ch 6] [30].

2.2.4 Positioning Applications

With positioning applications, a single fit-for-all type of solution doesn't exist and even a specific thought-out solution will most probably be compromised in some regard. Deliberating the performance parameters of a sensor and optics will help match the right optics and sensor with the application to reach as high performance as possible, aerial imaging as an example in [31].

In low-light environment applications where the amount of light reaching the sensor is vital and color information is not required, a monochromatic sensor with high pixel size and high quantum efficiency may offer the best performance. In general global shutter is preferable due to reduced distortions and artifacts in high-speed applications. For optics in positioning even though a popular 'fisheye' type of lens provides a large field-of-view, it may not be the optimal solution due to the limited resolution of the sensor and aggressive barrel distortion of the optics [32]. Higher framerate is generally better in high-speed applications [33]. Compared to passive depth captured by combining synchronized stereo cameras or neural-network-based monocular depth, active ranging [16] is more accurate and comprehensive, yet the range of depth sensing could be increased with visual sensors [34].

2.3 Inertial Sensing

Inertial sensors can be used to measure a system's linear and rotational changes [35, ch 2]. Linear change in system's inertia is caused by forces accelerating the mass of the system and can be measured with an accelerometer. Angular rate is measured with a gyroscope providing the changes in torque and thus the orientation of a system. Applications for inertial sensors are vast beyond positioning, a popular use case being gait analysis [36].

2.3.1 Gyroscopes and Accelerometers

Besides mechanical structures, modern gyroscopes can be implemented by utilizing optical or vibrating structures [35, ch 4]. Optical gyroscopes include Ring Laser Gyroscopes (RLG) and Fiber Optic Gyroscopes (FOG) both of which are based on Sagnac effect. Optical gyroscopes provide high accuracy while being large and expensive. Modern vibratory gyroscopes are manufactured to utilize the Coriolis effect on a vibrating element to track orientation. Various structures for accelerometers have been proposed and implemented, yet modern accelerometers utilize so-called solid-state structures, for example surface acoustic wave, vibratory, capacitive or piezoelectric [35, ch 6]. With low-cost accelerometers, the acceleration of a system is typically measured by a proof mass changing measurable properties of the system under acceleration, for example by changing capacitance between the proof mass and the mass' housing. Micro-Electro Mechanical Systems (MEMS) technology is generally used in manufacturing low-cost gyroscopes and accelerometers. MEMS sensors are less accurate than optical implementations yet physically smaller and cheaper.

An inertial measurement unit (IMU) is a device combining accelerometers and gyroscopes [37]. An IMU could also include a magnetometer, measuring heading based on the earth's magnetic properties. Determining position from acceleration information with ideal measurements would be to track position by the double integral of acceleration concerning time, the first integral being the velocity [38].

2.3.2 Performance and Error Sources

Inertial sensors can be divided into navigation, tactical, and automotive grades with declining accuracy respectively [37, 39]. An inertial sensor's typical sources of error appear as sensor biases and offsets, their variance, and noise introduced to the system via the sensor itself or its supporting components.

Performance parameters for an inertial measurement unit with an accelerometer and gyroscope are bias, bias instability, scale factor error, noise density, sensitivity and dynamic range. Sensitivity is a measure of the precision of the sensor and dynamic range the range of highest and lowest possible measurements. With inertial measurement sensors, error propagation is cumulative. Inertial measurement device's error sources can be divided into systematic and random errors. The systematic error includes sensor-dependent inaccuracies regarding for example manufacturing defects and device's integrated electronics. Long-term effects, such as cross-axis orthogonality and biases from temperature changes, can be compensated for with factory calibration. Bias or offset measures the constant offset of output without inputs. Similar constant bias is scale factor error with sensor measurements being offset by some constant factor. Bias instability is the change of constant bias over time resulting in drifting of offset. Random errors can be correlated or uncorrelated. Uncorrelated additive Gaussian noise yields from thermal noise and quantization noise in ADC. Similar types of error sources can be expected from other sensor types, such as from imaging sensor hardware and are effectively reduced by averaging the output. [37]

2.4 Satellite Positioning

Satellite positioning or navigation can provide highly accurate position by utilizing known timebase and orbital information of satellites transmitting data modulated within an L-band carrier radiowaves [40]. Satellite positioning systems are broadly referred to as Global Navigation Satellite Systems (GNSS).

2.4.1 Working Principle and Sources of Error

The general principle of satellite-based positioning is to calculate position from the known propagation rate of received radio waves, the satellite's position and common timeframe. The minimal case requires four satellites for calculating the 3DoF position and a time solution [40]. The first and most notable constellation is United States' Navstar Global Positioning System (GPS) followed by the Russian Global Navigation Satellite System (GLONASS), China's Beidou (Beidou-2/Compass) and European Union's Galileo. In addition to global positioning systems regional coverage satellite systems include Indian NavIC and Japan QZSS. The constellations vary in number of satellites, orbits and inclination resulting in coverage differences as well as variation in number of simultaneous satellites visible to a receiver. Multiple frequency bands for carrier waves are used for positioning, for GPS L1 (1575.42 Mhz), L2 (1227.60 MHz) and L5 (1176.45). [41]

The satellite position solution is based on measuring distances between a minimum of four satellites with known positions and the receiver. Distances or true ranges to satellites can be determined by aligning the receiver clock to the satellites' common clock and utilizing the knowledge of propagation speed and time of signals sent from the satellites. The position of a receiver in an ideal case without any distortions would then be trilaterated point on earth based on euclidian distances to the satellites, the fourth satellite is used for clock synchronization. Due to various sources of delay to the signal, calculated distances are referred to as pseudo-ranges. The satellite tracking and clock alignment on the Navstar system utilizes Coarse Acquisition (C/A) codes modulated within the carrier wave. C/A code includes a 1023-bit long satellite-specific sequence broadcasted every millisecond. A satellite sends its position/orbit and clock information at 50 bits per second modulated on top of a carrier wave. Code-Division Multiple Access (CDMA) method is used to spread a single L-band for all of the constellation's satellites at the same time. [40]

The main sources of error with satellite positioning systems are the variations in atmospheric conditions, more specifically ionospheric and tropospheric delays, as well as multipath errors due to occlusions and reflections delaying the received signals. [42, 43] Satellites in view and constellation geometry affect the accuracy of the position solution with Dilution of Perception (DOP) representing how effectively satellites in the receiver's sight are spread across the sky with a more pronounced spread of satellites providing higher confidence on the position calculated from the pseudo-range measurements. As the satellite positioning relies on the accuracy of known timebases and orbital positions of satellites, the accuracy of both a satellite's clock and broadcasted position affects directly the position accuracy calculated by the receiver. Distortions in radio wave propagation on satellite systems are not constant due to continuously changing conditions of the atmosphere. With perception or arithmetic errors on satellite orbits or receiver clock synchronization, the quality of position solution decreases while uncertainty increases. [40] The receiver's hardware-level error sources are the accuracy of tracking C/A codes and aligning the receiver's clock to satellites. Modern receivers include hundreds of so-called channels for tracking and aligning C/A codes sent by satellites.

2.4.2 Advanced Satellite Positioning Methods

Assisted GPS (A-GPS) methods can be used in gaining knowledge of the receiver's rough location or timing as well as the satellite constellation, almanac and broadcasted data without demodulating information from the signals sent from the satellites. A cellular tower can broadcast its location making the initial position assumption more accurate and thus decrease the time for an initial position fix. A constellation's almanac provided over the network can be used to seek satellites that should be in a receiver's sight. The advantage of A-GPS is the quicker time-to-fix, while the accuracy of reaching a positioning solution is not necessarily improved. Satellite-Based Augmentation Systems (SBAS) on the other hand can improve accuracy, integrity, reliability and availability by correcting atmospheric distortions present in global satellite systems on a regional yet wide-area level. [44]

Non-satellite-based augmentation systems include ground stations for correction of atmospheric distortions. Differential GPS (DGPS) and carrier-based satellite positioning methods, e.g. Real Time Kinematic (RTK) with or without utilizing multiple L-bands can provide more accurate position solution by accompanying a receiver with a ground base station. In DGNSS the satellite-specific PRN code modulated into carrier-wave can be compared to a known position solution of a base station. With a base station close enough to the receiver, typically called a rover, the atmospheric conditions are similar enough so that the ionospheric and tropospheric errors are equivalent for the rover and the base station making distortion detection and correction possible by eliminating the atmospheric errors. Differential GNSS systems can result in an accuracy of tens of centimeters. RTK utilizes the information of the phase of the radio wave in calculating the local sub-centimeter position solution. The position solution on a global scale is determined by solving an integer ambiguity search problem i.e. the integer number of wavelengths from satellite to receiver, the full position solution being a number of full wavelengths with the segment of partial wave determined from the phase. Accuracy of RTK is sub-centimeter with an integer number of wavelengths can be determined, or accuracy of a sub-ten centimeter in float-point solution. The disadvantage of DGNSS and RTK methods is the need for local or augmented correction information from the receiver in a known position. Virtual-Reference Station (VRS) is an augmented correction source generated by combining correction information from a network of receivers to model systematic errors in the atmosphere. The method will drop the need for a local base station in around 15 to 35 km radius with an accurately known position. [42]

The receiver's accurate position can be also determined by post-processing with Precise Point Positioning (PPP) [45] or Post Processing Kinematic (PPK) [46]. In PPP a large set of measured signals from satellites is collected and the data is then compared to a set of other base stations. While PPP is not a real-time solution, it provides a convenient way of measuring a single point accurately, for example to be used as a base station for atmospheric corrections. DGNSS and RTK can be post-processed or used in real-time by broadcasting corrections from the base station or VRS to the rover. Broadcasting via the Internet can be done using Radio Technical Commission for Maritime Services (RTCM) defined messages over Networked Transport of RTCM via Internet Protocol (NTRIP) [47].

2.5 Multi-Sensor Systems

Multi-sensor system includes two or more mutually synchronized sensors. Fundamental properties of synchronized a multi-sensor system are a common timebase between sensors and known physical alignment and placement, i.e. known pose, of the sensors [48, 49]. The pose of a sensor within a system's common coordinate frame is referred to as the sensor's extrinsic parameters. Individual sensors' intrinsic parameters modeling distortions are sensor specific and can be determined with calibration processes, for example with calibration patterns for cameras [50, ch 6.3].

The synchronization between sensors requires quite ofter synchronization between the systems that run the individual sensors [51]. A way of synchronizing a set of devices could be to assign a master clock for all of the devices and use the master clock's time as a common timebase [52]. Due to varying levels of drift with the devices' internal oscillators, a set timebase needs to be continuously enforced. An example of a master clock approach is an ethernet-based IEE1588 standard Precision Time Protocol (PTP) [53]. PTP uses one device as the master clock source and provides accurate synchronized time between devices via ethernet by timestamping the timing signals directly at the ethernet controller. Another common approach is to use GPS time as a common timebase. As previously discussed, the 4-dimensional solution of a GPS offers accurate timing in addition to providing a receiver's position [54, 55].

3. SOLUTION OVERVIEW

The chapter presents an overview of a mobile positioning solution with two pairs of stereo cameras, an inertial measurement unit and synchronized satellite positioning. The emphases of the presented solution are the cost-effectiveness and accessibility of the hardware as well as a comparison of the development approaches.

3.1 Hardware Development

Development approaches for a complete positioning hardware solution could be categorized based on the involvement required for implementing a system, the cost of a finished solution, and the required level of integration. The main approaches with the highest differentiation are referred to in this thesis as homebrew, with the most involvement, and turnkey with the least.

With the homebrew approach, the target is to utilize individual sensors and design the supporting hardware, e.g. communication solutions between sensors and computation, in-house. This approach gives the developer control over the details of the hardware and the cost of hardware may be lower, yet the involvement with the development is significantly higher than with the turnkey approach. With the turnkey approach, the system or subsystems are acquired as complete purpose-built solutions. Involvement required in deploying the solution is low, while the cost is high and adaptability or control over implementation specifics is out of reach for the developer. The high cost may stem from a manufacturer's typical customer base's requirements and applications, low volume of specialized hardware and higher-end components.

A range of options exists between the previously defined categories by utilizing intermediate products in some areas of the end-product, for example with computation or sensor solutions. Utilization of intermediate products as a base for the solution is referred to in the following chapters as a development kit approach. Utilization of a single strictly defined approach may not be an option, for example if a turnkey solution has proprietary technology. Similarly, the homebrew approach may be the only option if technology is highly experimental and commercial solutions do not exist.

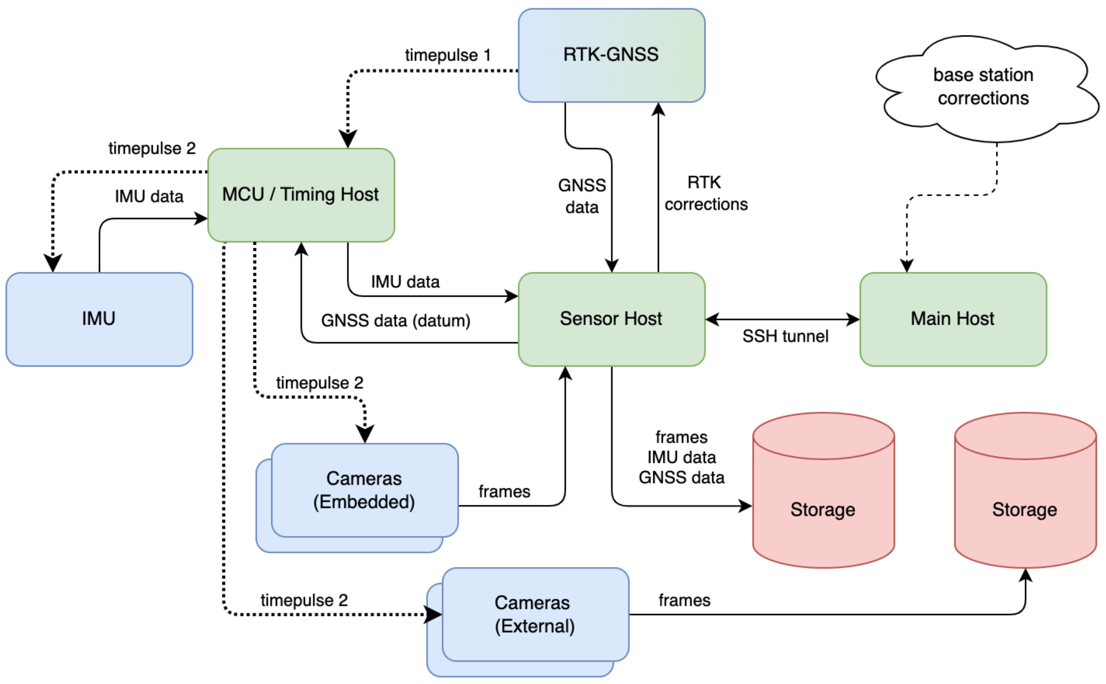


Figure 3.1. *System's operational design, dataflow and timing. Green blocks represent a computational subsystem, blue blocks sensors and red barrels storage. Continuous directional lines represent communication between systems, dotted line timing signals and the cloud a base station providing RTK corrections over cellular link.*

3.2 Implementation Overview

High-level requirements for a mobile positioning system's hardware are synchronization between sensors and reliability with the dataflow. Synchronization and reliability of a system are the bedrock for building robust position algorithms without the need for assessing constant errors from hardware sources within the algorithm itself. Overall system design is presented in figure 3.1 including the sensors and computation units.

3.2.1 Cameras

The choice for stereo camera pairs was driven based on required performance parameters, mainly the horizontal field-of-view of the lens/objective being over 100 degrees, framerate in the range of 60 to 120 fps, resolution of the sensor over 1 megapixel and a global shutter. The main differences between the two pairs in terms of initial planning were the baseline, i.e. the distance between the principal axis of the cameras, used. Pairs are referred to as embedded and external. The baseline is around 20 centimeters on the embedded pair compared to over 100 centimeters on the external pair. The external cameras represent a turnkey solution, embedded pair falls between the turnkey and homebrew approaches.

External cameras are a pair of FLIR Blackfly S USB3 with Sony Pregius IMX273 sensors

[56]. A monochromatic sensor is used compared to a color sensor for improved sensitivity. Sensor's 200 fps sustained framerate with 1.6-megapixel resolution are both beyond the specifications. The pixel size of 3.45 μm x 3.45 μm is larger than typical mobile sensors with reasonable resolution on a 1/2.9" format sensor. The USB3's 5Gbit/s transfer speed is required for the aforementioned resolution and framerate compared to quite typical 1-gigabit Ethernet cameras with lower comparative framerate, yet more industrial features, such as Power Over Ethernet (PoE). The key feature with most industrial cameras despite the used physical connector is the option for an external hardware trigger for easy hardware synchronization. The objectives chosen for the cameras are Theia Technologies SY110M [57] with 120-degree horizontal FoV with 1/2.5" format, around 110-degree horizontal FoV with 1/2.9" format sensor, and a focal length 1.7 millimeters. Theia objective corrects barrel distortion on the optics level, which may be beneficial due to less distortion and more light for individual pixels. Lens mounts into the camera with a CS-type mount.

Embedded cameras are a pair of monochromatic OmniVision OV9281 [58] by ArduCam. Resolution of 1 megapixel and framerate of 120 fps are within the preferred specifications. Pixel size and sensor size are both smaller compared to the industrial solution, with 3 μm x 3 μm and a 1/4" format. An M12 lens is used with the sensor 110-degree horizontal FoV. An external hardware trigger can be used with the OV9281. Arducam offers the camera and drivers with Raspberry Pi modules as well as Jetson Nano and Xavier NX, the Xavier NX being the most capable. ArduCam offers a kernel driver for cameras.

3.2.2 Inertial Measurement Unit

Integrating an inertial measurement unit into a larger set of sensors in the temporal domain requires either the ability to control the sampling timing of an IMU or IMU's supporting system to be able to provide correct timing signals for all of the other sensors in the system. Providers for turnkey solutions exist, for example VectorNav [59] and iXBlue [60] offering high accuracy hardware and software as a product with the possibility of system integration with external devices. Due to the low cost being one of the main targets of the implementation turnkey solutions would end up being relatively expensive, especially compared to the implementation's other components.

The development kit and homebrew approaches can provide significant decreases in the cost of hardware with lower accuracy consumer-grade IMUs and offer more variety while requiring the integration of software or software and hardware. The development kit approach would be ideal for a system in which an evaluation board's size and integrated microcontroller's available features, data transfer busses and expandability of the system are sufficient for the application.

In the case of this implementation, the cameras can be externally triggered while the GNSS timing is aligned to a top of a second with specified intervals between seconds

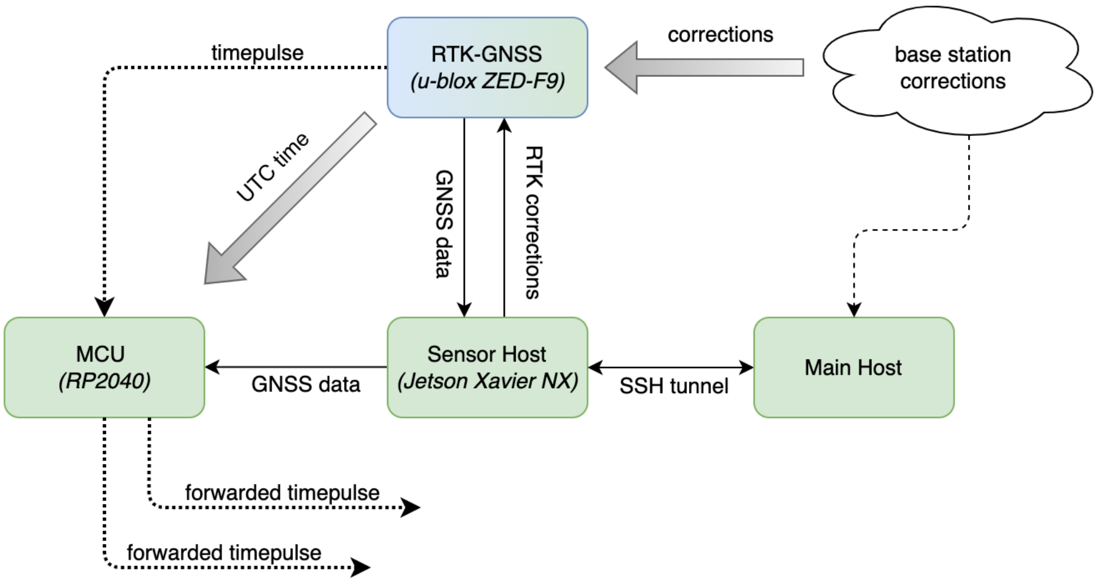


Figure 3.2. *GNSS host and host's integration with other subsystems. The base station corrections are requested in real-time by the sensor host via an SSH tunnel forwarded from the main host. The corrections are pushed to the RTK-GNSS sensor. The timing between all of the subsystems is handled by GNSS providing a rising timepulse for timing at the top of each UTC second with datum sent to the timing module via the sensor host.*

and provided as an external output, a timepulse [61]. The external timepulse from the GNSS can be utilized as a master clock for a common timeframe for IMU's supporting system to achieve high sampling frequency for the IMU as well as stepped-down timepulses for cameras. This functionality would be possible to implement with development kits, yet transferring raw sensor measurements to an external system with a commonly used interface would be problematic. For aforementioned reasons, the hardware was implemented from scratch as the homebrew approach with Murata SCHA600, also referred to as SCHA6XX, [62] series inertial measurement unit selected as the inertial sensor for the solution.

3.2.3 Satellite Positioning

In the right circumstances, i.e. clear view of the sky and a minimum number of occlusions, satellite positioning provides absolute scale and doesn't drift over time in contrast to pure visual-inertial systems. Using an RTK-capable GNSS sensor, the positioning solution error can be reduced to a sub-centimeter level making satellite positioning a viable solution for the ground truth. The RTK-GNSS sensor chosen is u-blox F9-series [63]. Other viable options would have been for example Swift Navigation [64], Serpentrio [65] or in terms of turnkey solutions RTK-GNSS products for construction sites e.g. Leica Geosystems [66]. The chosen solution is the most cost-efficient without drawbacks in terms of operational performance. The antenna used is an ANN-MB batch antenna from u-blox [67].

The general structure of the GNSS solution prosed in this thesis is presented in figure 3.2. The base station corrections used in the solution are provided by external parties, either with local base stations or with virtual stations (VRS). The communication with the base station is handled over a cellular link forwarded from the main host to the sensor host making it possible to run the sensor host individually without the main host by attaching cellular connectivity without reassigning any operational blocks of the system.

3.2.4 Supporting Systems and Computation

Supporting or interchangeably host systems include all subsystems required to run the sensor hardware. Available solutions for supporting systems are vast and the choices, in this case, come down to supported sensor hardware and available sensor drivers. The 3.3 shows all three hosts and the interfaces between them.

The computation should be sufficient for driving the sensors, transferring sensor data via bus into short-term memory and optionally storing raw or modified data in addition to running methods on top of data. Depending on the type of implementation in developing mobile positioning systems the integration with software may be already provided by the manufacturer of a sensor. Middlewares and frameworks such as Robotic Operating System (ROS) make it possible to deploy sensors without implementing the interface or Data Distribution Service (DDS) from scratch. Even though the immediate need for aggressive distribution of the required computation might be irrelevant, the scalability and adaptability of distributed computing and edge solutions will make the system and its parts more usable in the future. That being said, overengineering distribution of computation or data transfer for individual subsystems in the prototyping phase might be a worse option than utilizing middleware with relatively easy integration.

The three host systems of this solution are main, sensor and timing. The initial startup sequence includes starting up the devices and software, checking connections and setting global timing between devices. The global timing sequence is presented in figure 3.2. After the GNSS sensor has reached fixed status and the timepulse is active, the Coordinated Universal Time (UTC) is assigned to a specific timepulse by sending the UTC via sensor host into timing which manages systemwide timing. The timepulse is accurate to a maximum of a few hundreds of nanoseconds considering the delays with antenna, pulse generation and timing's interrupt to pulse [61]. This secondwise pulse is timestamped and matched with UTC top of the second time. The delay from the main host receiving the UTC information via UART connection and forwarding it to timing is nearly constant and less than the pulse interval which makes matching timing possible. After the initial match, the clock difference between the sensor host and timing host is known and can be monitored by capturing continuously UTC and timestamping timepulses.

After the initial start, the responsibilities of the main host are:

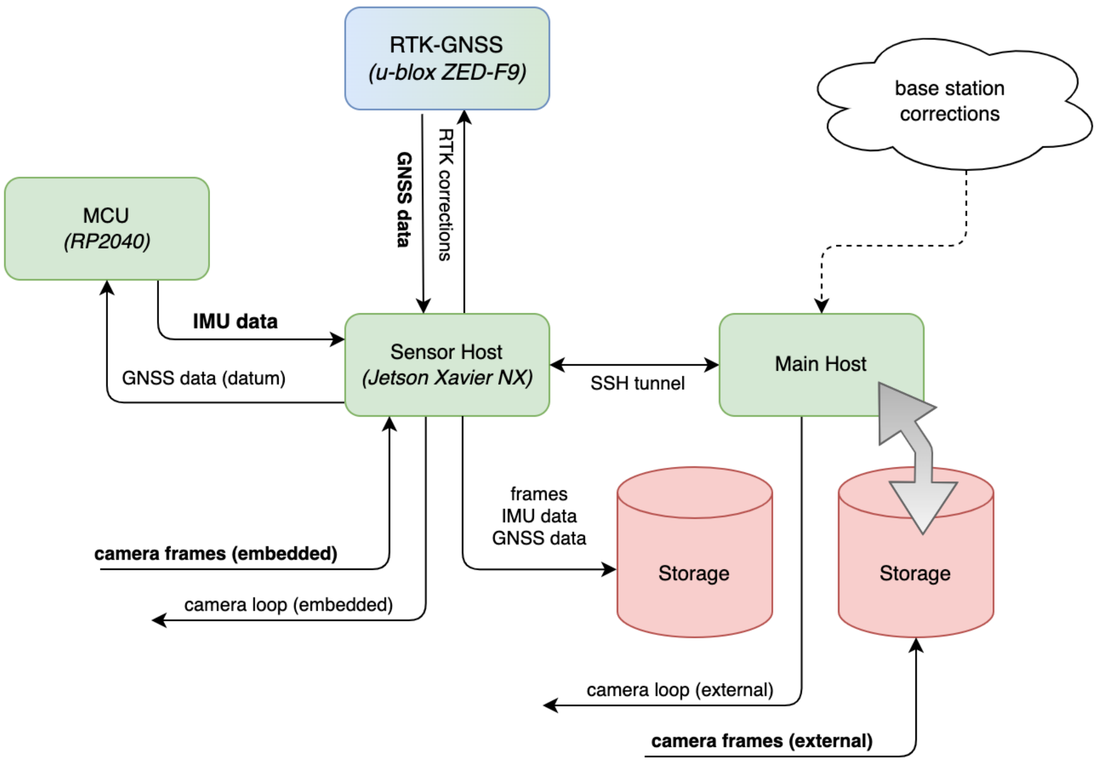


Figure 3.3. *Supporting subsystems. Directional sensor data as well as the sensor drivers and other software running on the host devices presented.*

- managing the cellular connection and providing port forwarded SSH tunnel for sensor host and
- running the capturing loop for the external cameras and accessing mass storage in which the frames from external cameras are written after acquisition.

The responsibilities of the sensor host are

- managing and running capturing loop for embedded cameras and storing the frames from the cameras into mass storage,
- capturing and storing inertial data coming from timing host and
- running the NTRIP client requesting RTK corrections and forwarding corrections while capturing incoming data from the RTK-GNSS sensor.

The responsibilities of the timing host are

- communicating with the IMU,
- forwarding raw timestamped IMU data to sensor host and
- hardware triggering camera pairs simultaneously with timepulse from GNSS as a master clock

4. SOLUTION IMPLEMENTATION

The chapter goes through the implementation of the positioning solution presented in the chapter 4. Technical drawings of the implementation can be found on Appendix A.

4.1 Cost-Efficiency

The Estimated Bill-of-Materials (BOM) is presented for the Integrated System in table 4.1 and External Vision System in table 4.2. As stated on the footnote rounded MSRP prices are presented due to the shortages of some components increasing the prices. The direct hardware cost is presented without the cost of development in time and equipment required.

The cost of External System's stereo cameras and optics are significantly higher than the integrated systems. Both stereo camera solutions provide the features required for this implementation's positioning solutions, more specifically hardware triggering and high enough framerate and resolution. The difference in performance in positioning applications would most likely come down to the baseline of cameras and some edge cases, for example machine vision camera's sensor being more sensitive in low-light environments.

4.2 Machine Vision Cameras

The large baseline stereo cameras used in the system are FLIR Blackfly S USB3 camera with Sony Pregius IMX273 sensors [56, 68] and Theia Technologies SY110M objectives [57]. This section covers the basics of GenICam machine vision standard, components required for a vision application, implementation of the software solution and hardware mounting.

A typical camera application includes device discovery and enumeration, control over the camera's initial and runtime settings and parameters, capturing and transferring frames over a link via some protocol for further processing and freeing the resources on the camera and application when the capturing process is finished. With embedded applications device discovery and enumeration are redundant in most cases, yet in generic machine vision applications enumeration is used due to differences in physical interfaces. The standard API widely used in the field of machine vision and industrial cameras is Euro-

Table 4.1. *Bill-of-Materials Estimate for Integrated System*

item	type	quantity	cost[*]	total
ArduCAM OV9281	CAM	2	50	100
ArduCAM lens	CAM	2	–	–
				100
u-blox ZED-F9R [1]	GNSS	1	300	300
u-blox ZED-F9P [1]	GNSS	1	250	–
u-blox ANN-MB	GNSS	1	–	–
				400
Murata SCHA6XX	IMU	1	120	120
Custom PCB [2]	IMU	1	30	30
				550
NVIDIA Jetson Nano [1]	COMP	1	150	–
NVIDIA Jetson Xavier NX [1]	COMP	1	400	400
				950
Power, Cables, Enclosure [3]	–	–	150	150
Mass Storage	–	–	–	–
				1100

[*] prices presented as rounded MSRP in EUR due to price fluctuations
[1] alternative product
[2] components and a single PCB without the IMU, total cost of producing a prototype set is higher
[3] material cost, fasteners and other hardware required

Table 4.2. *Bill-of-Materials Estimate for External System*

item	type	quantity	cost[*]	total
FLIR BFS-U3-16S2M-CS	CAM	2	400	800
Theia Technologies SY110M	CAM	2	250	500
				1300
Cables, Enclosure [1]	–	2	50	100
				1400
Computation [2] [3]	COMP	1	1100	1100
NVIDIA Jetson AGX Xavier [3]	COMP	1	800	–
Mass Storage	–	–	–	–
				2500

[*] prices presented as rounded MSRP in EUR due to price fluctuations
[1] material cost, fasteners and other required hardware
[2] computation unit used in this implementation
[3] alternative product

pean Machine Vision Association's (EMVA) GENeric Interface for CAMeras (GenICam) [69]. GenICam standard defines the interfaces, abilities via Standard Features Naming Convention (SFNC) and representation (GenAPI) between hardware level drivers and capabilities of specific hardware making it flexible for a developer to use any GenICam compliant software for controlling cameras across multiple camera manufacturers and models. Controlling the camera and capturing frames is done over physical interface standards, e.g. GigEVision or USB3Vision. GenICam includes GENeric Transport Layer (GenTL) for camera enumeration, data transfer and streaming without the need for manufacturer-specific implementations on for example the structure of transferred data, also proprietary implementations could be used in place of GenTL. GenTL producer running on camera hardware produces data for a consumer i.e. an application over any supported physical standard. FLIR provides a Spinnaker SDK [70], an extension of GenICam, utilized in this implementation including camera drivers and APIs for developing applications for FLIR cameras [71]. In addition to providing flexibility in combining hardware and software, standardization of the software components will generate reusable code by default which could further decrease the cost of developing and adapting existing systems for new hardware.

In terms of FLIR's product lineup USB3Vision and GigE cameras can utilize GenTL as well as wrapper APIs provided by FLIR [72]. The tradeoff between USB3Vision and regular gigabit GigEVision is the bandwidth and cable reach. USB3 offers up to 5 gigabits per second transfer speed over five meters of cable while GigEVision cameras' reach is tens of meters with a lower one gigabit per second bandwidth. For this implementation higher framerate and thus higher transfer speeds are a priority while the limited reach of the cables is not an issue. The bandwidth is higher with five or ten gigabits per second GigEVision cameras, for example with the FLIR Oryx 10GigE series, yet the camera hardware and supporting systems such as 10-gigabit switches offer limited options. The required bandwidth and speed of storage for continuous capturing can be calculated with frame bit-depth and channels i.e. bits required for each pixel, a number of pixels and framerate of a sensor accounting for additional headroom for protocol-related data. 12-bit pixel depth, 1440 pixels wide and 1080 pixels tall sensor and framerate of 227 frames per second produces

$$(12 \text{ bits} * 1440 \text{ pixels} * 1080 \text{ pixels} * 227 \text{ frames})/10^9 \text{ bits} = 4.24 \text{ Gbit/s}$$
$$4.24 \text{ Gbit/s}/8 = 530 \text{ MB/s}$$

about 4,24 gigabits or 530 megabytes per second per sensor raw image data without compression. Single USB3 interface's 5 gigabits per second and modern SSDs ability to handle well over a gigabyte per second sustained sequential write speeds make writing uncompressed images into the storage possible. Lowering the framerate to the previously set target of 120 fps results in continuous dataflow of around 2.4 gigabits or 280 megabytes per second.

Implemented application is built on top of the Spinnaker SDK. Spinnaker offers all of the required components and access to APIs using C++ as well as the existing codebase for developing and evaluating FLIR's camera hardware. The application supports multiple cameras. Hardware triggering for simultaneous frame capturing is enforced on the application level with captured frames being saved into RAW format. The capturing application follows the workflow presented at the beginning of this section. First connected cameras are discovered and enumerated, camera controls are set, frame capturing and simultaneous writing into SSD is started and then stopped manually or if frame trigger times out followed by cleaning up the resources. The camera setup can be accessed when starting the program, yet the need for accessing the setup is not necessary due to the ability to save preferences into the cameras permanently. Spinnaker SDK offers binaries to ARM-based systems as well as x86 [70], making it possible to run the capturing application on an ARM-based embedded platform, such as Jetson Xavier NX or Jetson Nano. The limiting factor with lower-end products would be the speed of the SSD or the computing required to process frames to compress frames and lower the required writing speed without hardware acceleration. Due to component shortage, a laptop with a high-speed SSD is used to run multiple processes on capturing software for both cameras. The application is available on github.com/kaarr/CE-HW-resources.

4.3 Embedded Cameras

The embedded cameras in this implementation are a pair of OmniVision OV9281 [58] sensors manufactured by Arducam. Arducam produces the carrier board for the sensor and the Video For Linux (V4L2) [73] kernel driver. The Arducam modules are attached to a Camera Serial Interface (CSI-2) on Jetson Xavier NX [74] [75] defined by Mobile Industry Processor Interface (MIPI) alliance [76]. CSI-2 Camera Physical Layer (C-PHY) and Display Physical Layer standards (D-PHY) include a 15-pin interface with four differential output data lanes, an input clock pair and an I^2C interface for configuration as well as power, ground and indicative signals in a Flat Flex Cable (FFC). The bitrate of a MIPI bus is defined by the lanes in use and the clock frequency provided by the camera host. Lanes are scalable i.e. the data can be divided simultaneously into all available lanes. Low-Level Protocol (LLP) enables differences in sent packet size with start and end of transmission flags. Packets have headers with data-id for virtual channels, word count and Error Correction Code (ECC) at the beginning and a footer with a checksum for error detection at the end.

The application is on top of the V4L2 kernel program which detects and displays the cameras as local system devices with individual /dev file handles. Cameras are configured with v4l2 over the I^2C interface while the frames are externally triggered from the timing module. The camera is set to wait for the trigger pulse, after which the data is sent to the host via CSI-2. OV9281 sensor's 10-bit 1280 pixels wide and 800 pixels high frame at 120 frames per second requires a bandwidth of

$$(10 \text{ bits} * 1280 \text{ pixels} * 800 \text{ pixels} * 120 \text{ frames})/10^6 \text{ bits} = 1228.8 \text{ Mbit/s}$$
$$1228.8 \text{ Mbit/s}/8 = 153.6 \text{ MB/s}$$

for raw image data. The total datarate is higher including protocol information. Given the bandwidth of 800 Mbps per lane with four lanes and 1000 Mbps on one to three, a clock of 1000 Mhz is adequate with Arducam's 2-lane solution. Similarly when storing the raw frame in a mass storage minimum of 153.6 MB/s per camera continuous sequential write speed is required. Continuously captured frames could be compressed and after the compression written into the mass storage to reduce the size of capturing if the mass storage would be the bottleneck of the system performance. Wrappers for V4L2 capturing and direct compression exist, OpenCV has a capturing and video writer implementation as an example. GStreamer includes multiple codecs with hardware acceleration for Jetson devices [77]. NonVolatile Memory Express (NVMe) Peripheral Component Interconnect Express (PCIe) 3rd generation four-channel Solid-State Drive (SSD) used with Xavier NX for storing RAW frames. The V4L2 application is available on github.com/kaarr/CE-HW-resources.

4.4 Timing Module and IMU Controller

Tasks assigned to a timing module are communicating with the IMU, forwarding sensor data to the host as well as initializing and keeping a common timeframe for all of the subsystems and sensors. Approaches for integrating IMU into the system were discussed in the previous chapter, implementing the IMU and computation from the scratch ended up being the approach with the most freedom, cost-effectiveness and scalability. The main steps for successful hardware implementation with the requirements presented in this thesis could be:

1. choosing an inertial measurement unit based on the application and evaluation of sensor on development or evaluation platform
2. selecting a microcontroller and supporting components to be used on the end product based on the system requirements
3. sensor driver implementation and validation on development platform or bread-

board

4. drivers for communication over the chosen bus with specific MCU and software for communication on the development platform or breadboard
5. custom hardware i.e. printed circuit board design, implementation and validation

The selected Murata SCHA6XX [78] uses two Serial Peripheral Interfaces (SPIs) for communicating with individual Application Specific Integrated Circuits (ASICs) UNO and DUE presented on [62]. DUE provides gyroscope data over the Y and Z axis, and UNO controls the X axes of the gyroscope and all axes of the accelerometer. Murata offers a development platform [78] with sensors on carrier boards as well as software with device drivers making the evaluation of sensor hardware convenient. Murata's development platform includes an STM32 Cortex-M4 microcontroller with programming and communication over Future Technology Devices International Limited's (FTDI) USB chip with Transistor-Transistor Logic (TTL) i.e. Universally Asynchronous Receiver/Transmitter (UART) serial communication link. This section will go over implementing sensor drivers and considerations of the data transfer options, then go over the implemented hardware and software. The software and hardware components are available at github.com/kaarr/CE-HW-resources.

4.4.1 Sensor Driver

Murata provides sample programs [78] for running SCHA600 on their STM32 development platform which will be used as a starting point for the development of sensor drivers on other platforms. The software components requiring modifications for developing sensor drivers for alternative platforms are the SPI communication with the sensor and hardware timers.

SPI utilizes four wires in a full-duplex mode; chip select (CS or SS), serial clock (SCK), input or Master (Controller) Out Slave (Peripheral) In (MOSI or COPI) and output Master In Slave Out (MISO or CIPO) [79, ch 6]. In full-duplex mode, MOSI and MISO lines can be used simultaneously to send and receive data. The working principle of SPI is based on one of the devices being a master which controls the chip select line and provides a clock signal (SCK) to all of the slave devices. If the master pulls the CS line down to logical zero i.e. near zero volts, the communication between master and slave devices has begun. The slave device doesn't affect when the line is down and can transfer data only when the data is requested by the master device. After the CS line has been pulled down master sends a request. The request has to be specific length, order and received by known clock polarity of the slave device. The polarity is represented by a clock polarity (CPOL) and phase (CPHA). Depending on the set CPOL the state of the clock signal is either high or low when CS is transitioning to low or high making it possible to unambiguously start data transfer simultaneously between master and slave devices. CPHA with mode

0 means that the data is sampled (MOSI) on each SCK's cycle on the rising edge and shifted (MISO) on the falling edge, as opposed to mode 1. [62, ch 5] In the case of Murata SCHA600 the length of a request and response is 32 bits, order most significant bit (MSB) first and polarity of the request mode 0 i.e. CPOL and CPHA are both 0. The physical SPI configuration for the sensor requires common SCK, MISO and MOSI lines with individual CS and EXT lines to be routed. This allows utilization of both communication interfaces sequentially by altering chip select signals between interfaces. A hardware timer is a timer running on MCU which compares increasing tick count, the number of clock cycles, to a comparative value in some register. When the value has been reached an interrupt is raised to notify the rest of the system about the timing threshold being reached and the timing register will be reset. The interrupt makes the microprocessor stash currently executable code into memory while the interrupt is dealt with, after which the execution of stashed code proceeds.

The implementation itself follows the code example from Murata on sensor registers [62, ch 6] and the initialization sequence of the sensor. Data structures used in capturing and transferring raw sensor data, beforementioned communication and timing as well as additional functionality differ from the example. The data capturing is managed in a buffer-based fashion in which the data is streamed to the host when a predefined buffer is full. The buffer size can be altered and offers flexibility as well as performance increase with the ability to send larger sets of data at a time, yet the possibility to stream data continuously in a FIFO manner with a buffer size of 1. The structure of the driver is based on the datasheet: initialize sensor with predefined startup sequence including setup of performance variables including sensor sensitivity and used filters, validate that the sensor is ready to be sampled and wait for requests to sample sensor. Both of the communication interfaces for UNO and DUE are set up in an interlaced fashion on the startup sequence [62, ch 4]. The startup sequence itself consists of retrieving predefined or generating new SPI frames which then are transmitted to the selected SPI device, with emphasis on the timing between transmitting frames having an impact on the setup of the sensor. The startup sequence yields a serial number of the sensor and flags determining if the sensor is set up correctly. In case of failure at the startup, a failure flag is set and returned. Accurate timing is kept with timers raising interrupts. Depending on the implementation and restrictions of the hardware and/or firmware of the microcontroller, sampling of the sensor could be done inside the interrupt in some systems, yet typically running large chunks of code inside the interrupt is considered a bad practice due to interrupt blocking other interrupts and limiting the functionality of for example timers. A safer way of utilizing hardware interrupts without losing the precision of timing is to set some sort of flag pointing to a code that will be run directly after exiting from the execution of an interrupt.

4.4.2 Data Transfer

Transferring data from a sensor via microcontroller to a host machine in a fast and secure manner with the ability to change the host machine is a key objective for the target solution. Options for data transfer protocols could be lower-level communication interfaces such as SPI, I^2C or RS232 [79, ch 6]. The problem with an embedded-level interface is the host's need for implementing a chosen interface. With SPI and I^2C physical interface has no standard connector, meaning that the bus itself can be easily adapted if the host machine has a General Purpose Input Output (GPIO) interface. I2C's typical transfer speeds of hundreds of kilobits to a few megabits per second are too low for solution's data transfer needs, while SPI's capability to transfer data with tens of Mbps, even close to a hundred Mbit per second would be adequate. Software on the other hand needs to be adjusted on each implementation on SPI accounting for the GPIO settings.

Options for higher-level interfaces include Universal Serial Bus (USB) or Ethernet. These communication protocols are fast with the USB3.0 base transfer rate of 480 Mbit per second and Ethernet Base100TX of 100 Mbit per second. For embedded systems transfer speeds over 100 Mbit per second are restricted by the microcontroller in most cases. Ethernet is widely used and can provide abstraction between microcontrollers and ethernet controllers, the cabling is widely used and can easily be converted to other widely used busses. The sent messages utilize Unified Datagram Protocol (UDP) to stream raw data from the sensor to the host as well as to communicate GNSS timing information with the sensor host. The format of messages is validated and standardized by utilizing protocol buffers. The communication between sensor and host requires static IP addresses and a common predefined gateway for routing requests between different subnetworks of the devices.

4.4.3 The First Prototype

The first prototype was based on the idea of utilizing ready-made and widely available carrier and breakout boards for stacking a unit with the required capability. The prototype is based on Arduino [80] platforms providing an integrated microcontroller with exposed GPIOs, an on-board programmer, power management with multiple options for power input and quite vast support for expanding the capability of a board with external 'hats' or 'shields' connected directly to the GPIO of a selected Arduino board. A custom PCB is created to connect the Murata SCHA6XX inertial measurement unit carrier board [81] to the Arduino platform.

The development was started with Arduino Mega [82] including a 16-bit AVR ATMega2560 microprocessor with a 16 MHz oscillator, multiple hardware timers, 54 digital GPIOs and compatibility with common shields, for example, ethernet shield. The downside of Arduino

Mega is the voltage level of 5 volts on the GPIOs requiring bi-directional level-shifters to communicate with the lower 3.3 volt IMU. Poor performance of the 16 MHz single core microcontroller was also a major disadvantage in developing an ethernet-based solution for data transfer on top of the communication with an IMU. Data being wrapped into consecutive frames to create a packet based on the OSI model requires a few milliseconds to be generated and transferred forward to the Wiznet ethernet controller without a large payload restricting the sampling rate of an IMU. Based on testing with Arduino Mega maximum sampling rate of the IMU considering data transfer delays would have been from 100 to 200 Hz i.e. from 10 down to 5 milliseconds per sample with a data buffer size of ten samples. Based on these issues with ATMega2560 more powerful Arduino Due [83] with 32-bit Atmel's ARM Cortex-M3 was selected as a follow-up platform, improving on all of the problems with the Mega platform. Even with Arduino Due having a single-core MCU, the increased performance could handle a higher sampling rate while retaining consistent data transfer. Utilizing Arduino libraries the software can usually be converted to other Arduino platforms without additional adaptation outside recompiling.

Ethernet hardware implementations are provided as shields to various Arduino platforms. The prototype utilized Arduino Ethernet Shield version 1 with a Wiznet W5100 ethernet controller [84]. The shield is mounted on top of the Arduino into the GPIOs from which the power delivery and SPI communication between devices are managed. Arduino provides an Ethernet library to be used with Arduino and an ethernet shield accelerating the prototyping process. An adapter PCB was designed for attaching the Murata IMU on top of the Ethernet Shield's GPIO extensions. Murata IMU on the carrier board is placed on the adapter board manufactured by a PCB prototyping company. The adapter itself is a double-sided FR-4 without copper pours or specified ground or power planes or any focus on signal integrity, Electromagnetic Interference (EMI) or ElectroMagnetic Coupling (EMC) [85].

All of the issues with the adapter, even though poor PCB design practices, were lesser issues compared to stacking multiple PCBs on top of each other leading to overall inefficient design in terms of the physical size of hardware, thus making enclosure design and production more challenging, as well as producing problems in expandability and scalability. Still as a proof of concept system and getting to know platforms, microcontrollers and components which could be used in following designs as well as validating custom driver for the IMU, considerations on manufacturing custom PCBs and integration with other sensors in the system the implementation was successful. In terms of initial design platforms like Arduino and the breadboard approach, assembling the system with DuPont/jumper wires on a breadboard can help assess the requirements of a design, validating functionality and estimating the stages needed and workflow for the final implementation.

4.4.4 The Second Prototype

The second prototype aims to fix issues and inefficiencies present in the first prototype. The goals for the second design are to consider a wider selection of microcontrollers and create a more integrated solution with more emphasis on lowering EMI and EMC with higher signal integrity. The ethernet controller from Wiznet is upgraded to W5500 [86] from the Arduino W5100 shield. The capabilities between chips are similar, despite the higher transfer speed of the new chip.

Microcontroller

The variety of microcontrollers from different manufacturers ranges in terms of performance, efficiency and cost among other parameters. Considerations on matching an MCU to an application could be

- software and hardware architecture: core-count, performance, including FPU, memory
- hardware features: communication interfaces, hardware accelerated features, peripheral support and substitutes (e.g. storage chips)
- existing codebase for drivers, compilers etc. software, community and speed of development, bare-metal or operating system implementations
- power requirements
- cost and availability of the MCU itself as well as components needed
- ease of expandability with new features, future-proofing

The priority of the list above varies depending on the application and compromises between the requirements and available MCUs have to be made. In this implementation the detailed requirements are running three operations: 1. communicating with the IMU, 2. transferring data over ethernet and 3. monitoring the timepulse signal from the GNSS master clock. These three operations can be executed sequentially on bare-metal, semi-sequentially with a threaded approach operating system, simultaneously with multiple processes running on a multi-core processor or some combination of the previous. The management of data doesn't require heavy computation within the MCU. Integrating timekeeping into hardware functions, such as hardware interrupts and timers can free timing operation into a threaded process without losing the accuracy of the timepulse. The IMU main program and ethernet communication could be done sequentially, similarly to the first prototype. With a higher-performance microcontroller, the delay when clearing the acquired data buffer and sending it forward would be shorter, meaning that the buffer could be larger with an acceptable delay. The sequential approach works well if the buffer size is small enough and unpredicted delays in data transmission won't occur. An option of communicating with the IMU within an interrupt set off by a hardware timer bypassing the data transmission and making it hold could be possible, yet is not considered a good practice and practically infeasible due to limitations on nested interrupts. Also, the IMU's communication interrupt would possibly be ceased by the timepulse interrupt which may be fine in practice, but expose the IMU communication to a non-constantly occurring delay in which the timepulse is handled.

A few issues need to be taken into account with the way of operating multiple cores. A bare-metal approach runs the application directly on the processor without any middleware handling operations. With a real-time operating system [87], such as FreeRTOS, the communication between cores as well as running multiple threads on a single core is made easy. The main problem with RTOSs in this implementation is the task watchdog. The watchdog is set to guard threads on a core at fixed intervals so that a single thread is not blocking all of the other threads thus exhausting the resources of the core and making tasks on other threads absent. The task watchdog assigns the computation time based on the priorities of the tasks, even with a single thread actively running. This could be avoided with critical sections which will block the watchdog as long as the critical section is executed, the critical section blocks also all of the other functionality of the operating system, such as hardware timers and interrupts making it a non-feasible solution. The communication between cores in a multi-core processor without an operating system can be done with shared memory and queues.

Based on the previous evaluation the RP2040 microcontroller from Raspberry Pi Foundation [88] was chosen. RP2040 has two 133MHz Cortex-M0+ cores without Floating Point Units (FPUs), the ability to run programs on bare metal, enough interfaces and support for needed features. Low power consumption of Cortex-M0+ was not a requirement yet would be beneficial in a scenario with very limited power. Another option would have been ESP32 from Espressif, yet the drawbacks of running RTOS on top of the dual-core hardware would have made development more complex.

PCB

The second prototype aims to improve the layered design of the first prototype into a more streamlined and spatially efficient design with an emphasis on good PCB design practices. The main requirements for the PCB are

- utilize RP2040 individually or on a carrier and integrated Wiznet W5500 ethernet controller
- limit the number of individual boards to a maximum of two; mother and daughter board
- the size of the board and its components is limited by the size of the enclosure

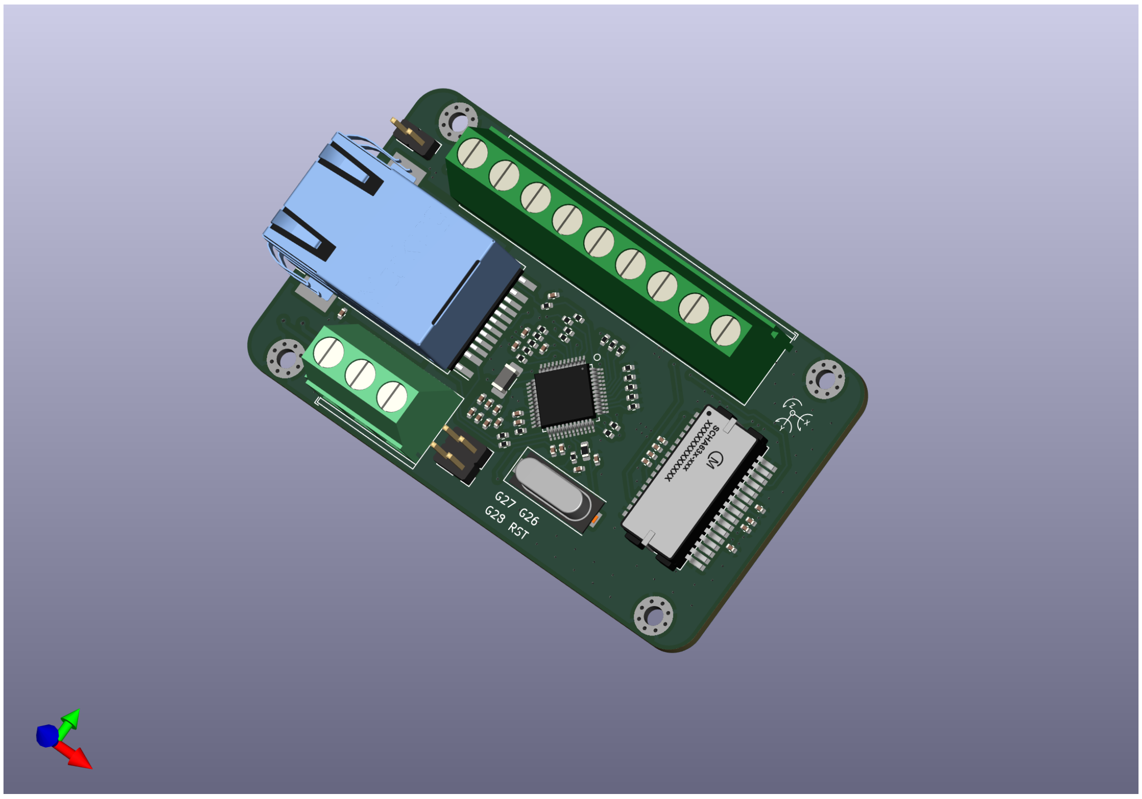


Figure 4.1. *Timing module render of the version 1.1, the RP2040 development kit is soldered into the backside of the board*

presented in A, the board will be placed behind the right camera and in a space of 8 (w) x 12 (l) x 6 (h) centimeters including the cabling required

- a screw terminal is required for incoming and outgoing timepulses
- power is sent to the MCU via either USB and/or power pins from 5V source, i.e. may need a buck converter
- rectangle pattern mounting points with a minimum of 2.5mm screw opening

Two designs with similar dimensions were created. In the first one, the IMU and ethernet controller were integrated into a daughter board run by a Raspberry Pi Pico RP2040 development board. The second one included only the IMU on the daughter board while the ethernet solution is integrated with the RP2040 on Wiznet's ethernet development platform W5500-EVB-Pico [89]. In both designs a four-layered PCB design is used. The layout has the bottom layers as power and signal, top power (1st design) or ground (2nd design) and signal and the middle layers are ground planes for improved EMI. The layers are stitched together with vias for low impedance power and ground routes as well as to reduce loop area. Critical signal vias are accompanied by a via closeby to tie fields into. Crosstalk has been reduced by the routing of the traces and introduction of ground traces with ground vias to couple energy to. The signal traces for communication are length matched to a few millimeters and a ground plane is positioned under the IMU with

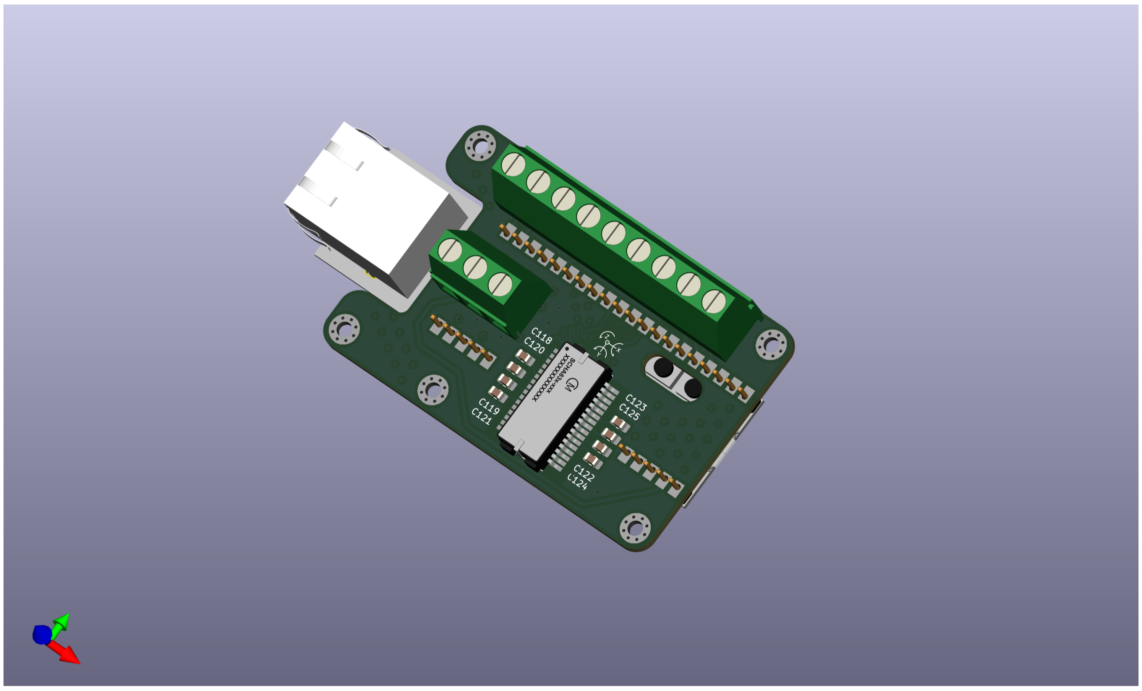


Figure 4.2. *Render of the timing module's version 1.2, the RP2040 and Wiznet development kit is soldered through pins visible on the board*

vias to the ground planes to layers two and three. The traces are not length matched for the timepulses on the screw terminal, the length matching would only be beneficial if the timepulse wires would be the same length [85, 90]. Murata's application note on Assembly [91] has been used as a reference for IMU's placement and fixing points. In the first design, the W5500 reference design has been used. The traces for the ethernet connector are impedance matched. The ground of a shield is connected to the internal ground via a capacitor. A set of uncoupling capacitors and pull-down/up resistors have been used where applicable. The motherboard, i.e. development board is attached via surface pads on the first and through a hole on the second design. Due to assembly, component or schematic error, neither of the implemented boards work. The design files are available on github.com/kaarr/CE-HW-resources.

4.5 Satellite Positioning

The approach of utilizing u-blox F9-series evaluation kits [92, 93] for the system's satellite positioning results in less involvement required with the integration process than with the previous section's IMU. Also, the focus of the implementation is more on the software and interfaces and setup of the evaluation kit as well as interfaces of the Xavier NX host device. Two different evaluation kits were used during the implementation, an evaluation kit for F9P [94] and F9R [63]. The differences between evaluation kits are mainly the carrier PCB, interfaces and integrated IMU for automotive dead reckoning in the F9R present in the final design. The accuracy of both F9P and F9R in RTK mode is sub-centimeter. Given an odometry input i.e. wheel tick and correct calibration sequence, the F9R's dead-reckoning capabilities could be used as another group truth source. Required components for the satellite positioning system are

- setting up the IO interfaces for sending and receiving data to and from the sensor and logging the sensor output
- implementation of a system for fetching corrections from the base station and providing those to the sensor
- developing a mounting solution for evaluation kit and antenna hardware considering possible interferences on kit and antenna

u-blox provides u-center evaluation software [95] for validating system performance. The software will be referenced in the following section, even though due to the limitations with the compatibility and features the software is not utilized in the final solution. Evaluation software from the manufacturer can significantly save time in development for evaluation and parts of the software, yet the licensing of the software and overall purpose for the software yielding closed-source modules limit the software usage in end-products.

4.5.1 Interfaces and Output

Required high-level interfaces for GNSS are presented in figure 3.2. The main interfaces concerning the u-blox are output and input from the sensor host and timepulse for the timing module. The timepulse signal can be accessed from a pin next to the F9R chip, for convenience, the pin is replaced with a single screw binging post, similar to a terminal block. Timepulse requires also common ground from the PIO (Programmable Input Output) pins on the rear 10-pin connector of the evaluation board, pin 13 [93]. As for the communication the F9R evaluation kit includes two UARTs, CAN, SPI, I^{2}C and USB interfaces, the most versatile interface for the solution's use case is to use USB for communication with the sensor as well as power delivery for the evaluation board.

The setup of interfaces [96] includes the setup of the interface itself and the setup of the input/output for that specific interface, both of which can be done via the u-center or manually by sending configuration commands via some serial interface. The commands for interfacing with the F9R are available on the ZED-interface description [97]. The sensor output includes messages from NMEA or UBX protocols, or the extension of NMEA (PUBX), defined in the interface description. The minimum messages required include position information with latitude, longitude and altitude, date information or the current time of the week in a known timeframe (e.g. UTC ToW) for matching timepulse with the date and positioning solution and timing as well as fix statuses. Additional information could be satellite related, e.g. satellites in sight, satellite constellations and Dilution of Perception (DoP) in three axes describing the spatial distribution of satellites for a position fix. The u-blox F9-GNSS sensors can output positioning solutions in higher precision and alternative coordinate systems as well as the estimated accuracy of calculated position solution with system-level data from for example power status. An input interface configuration needs to be set to accept RTCM messages matching the output of the base station's mount point.

4.5.2 Atmospheric Corrections and Antenna

RTK corrections are fetched from the base station over a network via NTRIP Client. As discussed before the base station provides the correction data over the network via an IP address, port and a mount point. The base station can be fixed or virtual. With a fixed base station, the rover has to be in the range of the station, typically from 10 to 35 kilometers. A set of base stations produce enough data to model errors in the atmosphere as a virtual base station for estimating error terms in an arbitrary position within the range of physical stations. The mount points are used to provide exclusive/inclusive sets of messages and message types regarding for example RF signal observations from different constellations. RTKLIB [98] is an open source package including RTK positioning algorithms, message protocols and communication protocol implementations for interfacing between systems. RTKLIB's str2str NTRIP client receives RTCM version 3 correction data from the base station and can be used to send the nearby position of a receiver for the generation of PRS/VRS.

The antenna used is u-blox ANN-MB patch antenna [67] requiring a ground plane. The size and shape of a ground plane affects the antenna performance [99, 100]. The mounting of the evaluation kit hardware has the same constraints and limitations as with discussed in the IMU section if the integrated IMU of the F9R is used. An issue concerning the antenna and interference of the radiation from USB3 should be taken into account. USB3's data spectrum's 1st half harmonic frequency is close to L-bands used with GNSS and can affect an antenna's reception of satellite signals on those bands [101].

4.6 Synchronization

An accurate common timebase could be implemented on top of data transfer protocols with purpose-built hardware by utilizing the IEE1588 Precision Time Protocol (PTP). With PTP a large set of devices could be synchronized, yet providing a timing signal or timepulse for a limited set of devices by a masterclock is a simpler approach without requiring PTP-supported ethernet controllers. An initial common timeframe and continuous timestamping of new data will also make tracking clock drift between systems possible. A timepulse-based implementation requires a startup sequence for a common timebase and temporal synchronization of the subsystems.

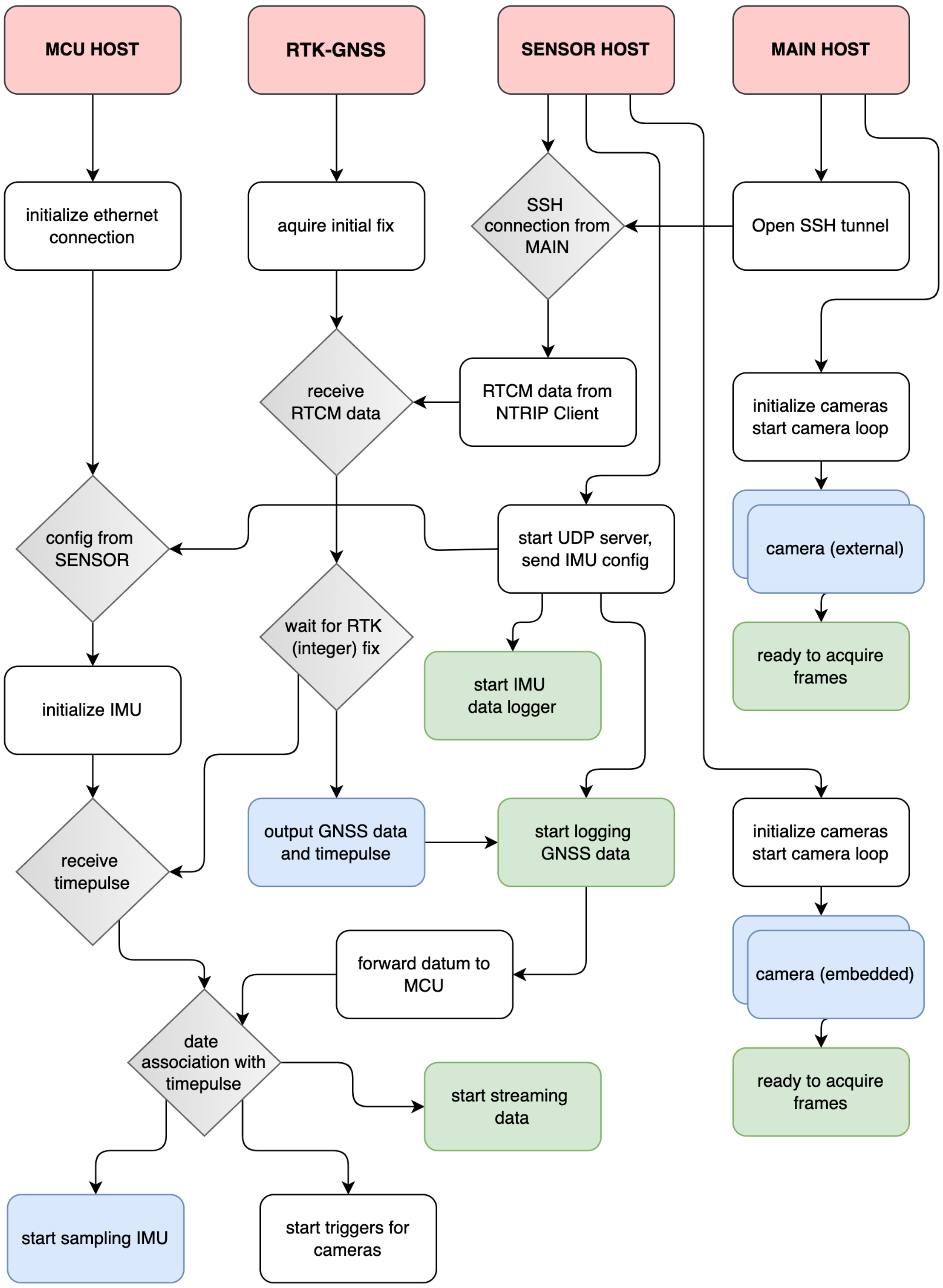


Figure 4.3. *Startup sequence and temporal synchronization, the subsystems are presented on the top row, green blocks represent finished startup for software components and blue ones for hardware sensors. Grey diamonds are non-bypass decision points when subsystem has to wait for another subsystem to provide a service.*

The startup sequence of the implemented system is presented in figure 4.3. The white boxes represent services required by either the device itself, for example sensor initialization steps, or another subsystem. Diamond shapes are blocking operations for waiting on another subsystem to start a service. Blue and green boxes are the hardware and software startup sequence finishing points after which acquiring synchronized data can be started. An absolute fixed initial time is acquired by the GNSS sensor providing an accurate timepulse for the timing module per second basis. The data connected to the timepulse is forwarded via the already established communication interfaces between the GNSS sensor and the sensor host, and the sensor host and timing module. After the timing host's clock offset compared to the timepulse is known, the initial synchronization has been reached. A continuous synchronization will be kept between the GNSS and timing host by continuously timestamping the incoming timepulses against the timing host's clock. This way missing timepulses can be discovered and the possible drift present on the timing host's clock source can be corrected.

4.7 Power Delivery

In this implementation, power delivery needs can be divided into powering machine vision cameras and powering integrated systems. The system's power budget is discussed, and implementation details and different powering options are presented and discussed for each subsystem.

Requirements for a power delivery system for machine vision cameras depend on the host system. Machine vision cameras can be powered from an external host over the USB or via Hirose HR10 [68]. Injecting power via a Hirose connector would be beneficial if the host machine connected via USB cannot provide consistent power, the host system's power output is limited, the host's power budget is already used or the host relies on battery power and an external power source can be used. Machine Vision cameras require 3 watts of power when running. With USB's 5 volts the current will be 600 mA, within USB3 power spec of 900mA per port. Power delivery via Hirose HR10 requires a voltage of 12V to 24V lowering the current required and ideal for applications with higher DC voltages available. Power delivery can be done with the host system's USB buses.

Possibility for portable power options as well as a wide voltage range is required for the integrated subsystem. The power consumption of individual modules is presented in table 4.3. Considering Xavier NX's power output of 900mA for each USB3 port, u-blox GNSS sensor and the timing module can be powered from Xavier's USB interface. The stereo cameras will be powered from the MIPI interface nonetheless. Xavier NX developer kit has a range of usable voltages, from which the core voltage and other required IO voltages will be stepped down. As an example, widely available battery options in the required voltage range of 9 to 20 volts are typical car batteries with around 12-volt native voltage

Table 4.3. *Power Budget of the Integrated Vision Subsystem*

	power options	**voltage ranges**	**consumption**
Xavier NX	barrel connector	9-20V	∼15W max in 6-core mode
ArduCam OV9281	MIPI CSI-2	IO 1.8V	134mW per sensor, ∼250mW per unit
u-blox F9R	USB, ext. power	5V, 5V-24V	3.3V @150mA max, ∼500-1000mW
IMU module	USB, ext. power	5V, 3.3V	W5500 ∼75mA @3.3V, RP2040 max 1500mW
max			∼18W

and more sophisticated lithium-based battery solutions. Lithium-ion cells can be found in ready-made packages with and without a battery management unit (BMS). Four lithium-ion battery cells with 18650 form factor in series are used in this solution attached battery holders due to the freedom of mounting holders and easily adaptable and changeable cells. The total capacity of the battery pack is 12 watthours per cell for four cells, i.e. 48Wh in total at an average 14.8 volts with four 3.7V cells in series. With a maximum power consumption of 18 watts, the battery solution has theoretical runtime 2 hours and 40 minutes.

4.8 Mounting and Enclosure Solutions

This section discusses the mounting solution used for each subsystem, the main subsystems being the roof rack for fixing all of the external enclosures as well as the enclosures for machine vision cameras and integrated satellite positioning, inertial measurement unit and timing module, stereo cameras and computation unit. Conceptual technical drawings for the solution are presented on A and design files are available at github.com/kaarr/CE-HW-resources.

4.8.1 Digital Manufacturing and Materials

The requirements for enclosures are splashproof and scalable design with effective heat transfer properties. The layout of a system disregarding synchronization comes down to specific hardware mounting, heat dissipation and application. Some standards exist for sensor mounting, standard screw pattern of machine vision cameras as an example, making it easier for manufacturers to build solutions on top of a wide range of products. Computer Aided Design (CAD) with additive manufacturing methods offer low-cost materials with enough strength and heat transfer capabilities. Material choices and structural design should have an emphasis on introducing as little flex as possible for keeping the spatial calibration intact.

The roofracks used for mounting the hardware into a car are a pair of Thule Square-Bar steelpipes connected with rivetted tieplates. The dimensions of the roofrack can be found in appendix A section C. Even though the chosen steel pipes are heavier and less aerodynamic, which are typical selling points of roof racks, the steel pipes do not flex similarly to aluminum and thus provide a more stable platform for fixing sensors in place. As stated before, preserving the spatial calibration of sensors during changing stresses applied to the mounting system is vital in keeping the calibration between sensors consistent. In addition to six on top and one on the bottom rivetted 2.5 millimeters thick tie plates the strength of the system is increased by the four mounting points in contact with either integrated or open roof rails. The roofrack fixing points are hooked on with a hinge mechanism which tightens the pipes in place simultaneously as it tightens individual rail mounts into a car's roof rails.

4.8.2 Machine Vision Cameras

The requirements for the enclosure and mounting The FLIR Blackfly S [102] is a so-called ice-cube form factor camera, a standard widely used with machine vision cameras. Because the form factor is standardized a mounting solution for the camera can be utilized with various sensors from different vendors. The changing part of this kind of mounting system between different cameras is the size and shape of the optics and field-of-view. The implemented solution is presented on appendix A section A. Driving factors for the design of the enclosure were the physical requirements as well as the reusability of the design. The enclosure consists of three bodies secured together with screws. The front body houses the shield for the optics and can be modified to fit different lenses or objectives. The camera itself is attached to the middle part with standardized mounting holes. The back body is mounted into the roof rack with three bolts going through the back part and connecting to mounting rails.

Cable pass-through is implemented with socket seals and printed cable mounts for the USB3 and Hirose HR10 cables. The main bodies are made from polylactic acid (PLA), the rails from carbon-fiber reinforced Markforged Onyx [103], the shield from 3-millimeter thick laser-cut acrylic and the gaskets from nitrile rubber (NBR). The decision behind using different materials concentrates on the sturdiness of the system in the difficult environment; PLA offers a sturdy mounting platform without unnecessary flexibility while the structure and number of mounting points of the more flexible and stronger Onyx in the rails will make sure that the camera mount is not going to disintegrate during high-speed driving. Infill factors for PLA parts make a difference in the waterproofing of the enclosure. The middle part is under the most stress, so an infill factor of 80 percent was used while 50 percent was adequate for the front and back parts. The shield could be more robust if polycarbonate was used, yet ultraviolet radiation from laser cutting discolors transparent polycarbonate making precision cutting polycarbonate more challenging with the tools available. The enclosure is splashproof. The design's ability to withstand harsh environments and continuous stress could be further validated with simulations and physical testing.

4.8.3 Integrated Vision System

The 'Integrated Vision System' includes integrated satellite positioning, inertial measurement unit, stereo cameras and computation unit. A one of design was initially made as a rough proof-of-concept with metal hardware, e.g. aluminum tubing and brackets, for keeping the calibration of the devices and a plastic box was installed over the metal mounting system for protecting the sensors. The main features required for a successful design

- rigid mounting for sensors
- as water and impact proof of an enclosure as possible with minimum material; socket seals for cable pass-throughs, seals between individual parts, preferably minimize the number of screws going through the enclosure and shields for cameras
- the enclosure has to be 3D printed on a printing bed the size of Prusa MK3 [104]
- assembly and fastening of the devices from outside or easily from inside considering the assembly order, interlocking mechanisms for non-critical sensor hardware, stacking/subframe/core approach for mounting non-critical devices
- enough mounting points for mounting the device into the roof racks
- scalable and modifiable design for substituting components, standard mounting hardware and enough buffer/slack for replacements
- easily replaceable battery solution and an option for using an external power source
- the enclosure should be able to dissipate generated heat, preferably without an active cooling

The enclosure consists of two mirrored sides. The computation devices, Xavier NX and u-blox evaluation board are mounted together with a stacking approach and the 'core' including those devices is interlocked into place when two sides of the enclosure are united. A key remark with individual components is that the space requirements should not only consider the devices themselves but the cabling needs to be accounted for as well. An individual mount for the IMU board can be printed into the enclosure piece or printed separately and epoxied on. Four cables, ethernet, antenna, and two Hirose HR10 cables for FLIRs, are passed through via seal clamps. An additional seal clamp is added to the battery cover housing the 18650 cells in Keystone 1049P battery holders for using an external power source. The battery cover is held on with six screws with nut counterparts secured inside the enclosure making assembly from outside possible, cover is sealed with a laser-cut NBR seal. The sides of the enclosure are fastened together with the printed reinforcement part and five outside screwpoints as well as the battery cover.

The ANN-MB GNSS antenna is fastened down with the ground plane with embedded nuts. Six through-hole mounting points are placed on the edge of the enclosure sides making the total number of screws going from outside to inside the enclosure four per camera and six for the batter cover. These 14 points are waterproofed with waterproof counterparts or silicone ring seals. Cameras are mounted with two points into the chassis and two points within the camera assembly including the shield and additional mounting hardware. Cameras are sealed with laser-cut seals. The devices can be fastened in place from the outside disregarding the IMU module which can be fastened with a 90-degree wrench from the right side chassis camera mounting hole or the battery cover mounting plane. Enough clearance for the cables and excess space for additional hardware. Devices inside can be substituted by only redesigning the individual mounting systems for each sensor, e.g. camera assembly or the computation core devices. PLA was used in all of the plastic 3D printed parts of the build, acrylic for the shields and NBR for seals. Cable guards are screwed on the roof racks for protecting the connectors.

The total power output of the system is assumed to be lost, the maximum power is around 20W assuming that the heat from battery cells is contained within 18W calculated maximum output with the main heat source being Xavier NX [105]. Xavier NX is not run with full load reducing the realistic heat load. The enclosure has some internal airflow due to included fan with Xavier NX's cooling solution. The heat transfer methods are radiation, and convection via enclosure's internal air. Disregarding the radiation, the convection can be roughly estimated with temperature rise inside the enclosure, i.e. the temperature in which the inside air will have the equilibrium point with outside air's temperature. The heat is convected through the inside walls of the PLA chassis on a thickness of approximately 3 millimeters without the structural reinforcements on the inside walls and the battery cover. For a worst-case approximation, a 10 millimeters average thickness of the walls will be used. The infill of the chassis parts is not 100 percent and can affect the thermal performance due to air being a superior insulator compared to solid plastic. Heat convection can be modeled as resistive elements in heat transfer: transferring heat to inside air and transferring heat from inside air into the interior walls and from interior walls into exterior walls. Internal temperature rise [106, ch 2] is calculated with the surface area of the enclosure, wall thickness, thermal conductivity and power generated by the heat source. 0.30 m x 0.15 m x 0.10 m enclosure makes the surface area of 0.18 m^2. with an average wall thickness of 10 millimeters and material PLA's thermal conductivity being around 0.2 watts per m^2 per Kelvin gives a temperature rise of 5.56 degrees Celcius inside the box based on modeling the equilibrium of a heat flux through a structure and outside temperature

The design could still be further improved. The fastening of the enclosure sides is not reinforced on the battery cover attachment points making it prone to leaking if the battery cover is not fitted correctly or there is not enough clearance with the battery cover's outer dimensions. The cameras should be fastened down to the chassis and the camera assembly with at least three screwpoints meaning six screws in total. In this design, two screws seem to offer a solid enough mounting solution, yet with a slightly curved chassis mounting plane, the cameras could move slightly breaking the calibration. Still, the mounting system for the 'Integrated Vision' devices seems to be sturdy, yet compact enough for using it on the roof rack system. The individual parts can be 3D printed on printer form factors widely available and additional hardware, such as seal clamps and screws is widely available and could be easily substituted.

5. CONCLUSIONS

This thesis provided an overview of visual, inertial and satellite positioning hardware and presented concepts and requirements for multi-sensor mobile positioning. Visual-inertial odometry was chosen as a cost-effective and high-performance specialization from different subsets of multisensor setups. An example implementation of a visual-inertial system with an integrated high-accuracy satellite positioning sensor was presented with thoughts on the design and development approaches. Practical details about physical interfaces, hardware integration, computation and the development processes of different approaches were presented.

The main target of the research was to assess the possibilities of implementing a synchronized hardware solution and determine the critical requirements of sensors, computation and supporting equipment. It is possible to integrate a cost-effective set of hardware into a synchronized system with most, if not all, key features of industrial so-called turnkey solutions and simultaneously lower the cost of the end product significantly. All hardware is and should not be created equal in terms of how the implementation process is approached. Producing the hardware from scratch, while possible with certain types of hardware, will require more development time than other approaches and thus increase the cost of the solution with one-of or proof-of-concept solutions.

As for future research, the performance of the implemented system should be tested by running positioning algorithms was not covered by this thesis. While some considerations on sensor properties and performance were presented, a real-world test would be required to determine the true performance of sensors compared to each other, i.e. homebrew solution's sensors compared to an industrial solution, and ultimately compared to methods utilizing more expensive sensor hardware, LIDAR as an example.

REFERENCES

[1] Yuan, S., Wang, H. and Xie, L. Survey on Localization Systems and Algorithms for Unmanned Systems. *Unmanned Systems* 9.2 (Apr. 2021), pp. 129–163. ISSN: 23013869. DOI: 10.1142/S230138502150014X.

[2] Bresson, G., Alsayed, Z., Yu, L. and Glaser, S. Simultaneous Localization and Mapping: A Survey of Current Trends in Autonomous Driving. *IEEE Transactions on Intelligent Vehicles* 2.3 (2017), pp. 194–220. ISSN: 23798858. DOI: 10.1109/TIV.2017.2749181.

[3] Singandhupe, A. and La, H. A Review of SLAM Techniques and Security in Autonomous Driving. *Proceedings - 3rd IEEE International Conference on Robotic Computing, IRC 2019* 19 (2019), pp. 602–607. DOI: 10.1109/IRC.2019.00122.

[4] Yurtsever, E., Lambert, J., Carballo, A. and Takeda, K. A Survey of Autonomous Driving: Common Practices and Emerging Technologies. *IEEE Access* 8 (2020), pp. 58443–58469. ISSN: 21693536. DOI: 10.1109/ACCESS.2020.2983149. arXiv: 1906.05113.

[5] Alkendi, Y., Seneviratne, L. and Zweiri, Y. State of the Art in Vision-Based Localization Techniques for Autonomous Navigation Systems. *IEEE Access* 9 (2021), pp. 76847–76874. ISSN: 21693536. DOI: 10.1109/ACCESS.2021.3082778.

[6] Li, X., Yi, W., Chi, H. L., Wang, X. and Chan, A. P. A critical review of virtual and augmented reality (VR/AR) applications in construction safety. *Automation in Construction* 86 (Feb. 2018), pp. 150–162. ISSN: 0926-5805. DOI: 10.1016/J.AUTCON.2017.11.003.

[7] Sahu, C. K., Young, C. and Rai, R. Artificial intelligence (AI) in augmented reality (AR)-assisted manufacturing applications: a review. *International Journal of Production Research* 59.16 (2020), pp. 4903–4959. ISSN: 1366588X. DOI: 10.1080/00207543.2020.1859636.

[8] Liu, Y., Fu, Y., Chen, F., Goossens, B., Tao, W. and Zhao, H. Datasets and Evaluation for Simultaneous Localization and Mapping Related Problems: A Comprehensive Survey. (2021). arXiv: 2102.04036. URL: http://arxiv.org/abs/2102.04036.

[9] Thrun, S. *Probabilistic robotics.* 2005, p. 647. ISBN: 978-0-262-20162-9.

[10] Nistér, D., Naroditsky, O. and Bergen, J. Visual odometry. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* 1 (2004). ISSN: 10636919. DOI: 10.1109/CVPR.2004.1315094.

[11] Yousif, K., Bab-Hadiashar, A. and Hoseinnezhad, R. An Overview to Visual Odometry and Visual SLAM: Applications to Mobile Robotics. *Intelligent Industrial Systems 2015 1:4* 1.4 (Nov. 2015), pp. 289–311. ISSN: 2199-854X. DOI: 10.1007/S40903-015-0032-7.

[12] Durrant-Whyte, H. and Bailey, T. Simultaneous localization and mapping: Part I. *IEEE Robotics and Automation Magazine* 13.2 (June 2006), pp. 99–108. ISSN: 10709932. DOI: 10.1109/MRA.2006.1638022.

[13] Särkkä, S. *Bayesian Filtering and Smoothing*. Cambridge University Press, Jan. 2013, p. 232. ISBN: 9781107030657. DOI: 10.1017/CBO9781139344203.

[14] Li, X., Chen, W., Chan, C., Li, B. and Song, X. Multi-sensor fusion methodology for enhanced land vehicle positioning. *Information Fusion* 46 (Mar. 2019), pp. 51–62. ISSN: 1566-2535. DOI: 10.1016/J.INFFUS.2018.04.006.

[15] He, X., Pan, S., Gao, W. and Lu, X. LiDAR-Inertial-GNSS Fusion Positioning System in Urban Environment: Local Accurate Registration and Global Drift-Free. *RemS* 14.9 (May 2022), p. 2104. ISSN: 2072-4292. DOI: 10.3390/RS14092104.

[16] Choi, J. Range Sensors: Ultrasonic Sensors, Kinect, and LiDAR. *Humanoid Robotics: A Reference* (2017), pp. 1–19. DOI: 10.1007/978-94-007-7194-9_108-1.

[17] Labbé, M. and Michaud, F. RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation. *Journal of Field Robotics* 36.2 (2019), pp. 416–446. ISSN: 15564967. DOI: 10.1002/rob.21831.

[18] Usenko, V., Demmel, N., Schubert, D., Stuckler, J. and Cremers, D. Visual-Inertial Mapping with Non-Linear Factor Recovery. *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 422–429. ISSN: 23773766. DOI: 10.1109/LRA.2019.2961227. arXiv: 1904.06504.

[19] Campos, C., Elvira, R., Rodriguez, J. J., Montiel, J. M. and Tardos, J. D. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial, and Multimap SLAM. *IEEE Transactions on Robotics* 37.6 (2021), pp. 1874–1890. ISSN: 19410468. DOI: 10.1109/TRO.2021.3075644. arXiv: 2007.11898.

[20] Seiskari, O., Rantalankila, P., Kannala, J., Ylilammi, J., Rahtu, E. and Solin, A. HybVIO: Pushing the Limits of Real-time Visual-inertial Odometry. *Proceedings - 2022 IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2022* i (2022), pp. 287–296. DOI: 10.1109/WACV51458.2022.00036. arXiv: 2106.11857.

[21] Gallego, G., Delbrück, T., Orchard, G., Bartolozzi, C., Taba, B., Censi, A., Leutenegger, S., Davison, A., Conradt, J., Daniilidis, K. and Scaramuzza, D. Event-Based Vision: A Survey. (2022). DOI: 10.5167/UZH-185139.

[22] Gade, R. and Moeslund, T. B. Thermal cameras and applications: a survey. *Machine Vision and Applications 2013 25:1* 25.1 (Nov. 2013), pp. 245–262. ISSN: 1432-1769. DOI: 10.1007/S00138-013-0570-5.

[23] Mueggler, E., Rebecq, H., Gallego, G., Delbruck, T. and Scaramuzza, D. The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM. *Sage Journals* 36.2 (Feb. 2017), pp. 142–149. ISSN: 17413176. DOI: 10.1177/0278364917691115. arXiv: 1610.08336.

[24] Nakamura, J. *Image sensors and signal processing for digital still cameras*. 2017, pp. 1–336. ISBN: 9781420026856. DOI: 10.1201/9781420026856.

[25] Oike, Y. Evolution of Image Sensor Architectures With Stacked Device Technologies. *IEEE Transactions on Electron Devices* 69.6 (June 2022), pp. 2757–2765. ISSN: 15579646. DOI: 10.1109/TED.2021.3097983.

[26] Durini, D. *High performance silicon imaging : fundamentals and applications of CMOS and CCD sensors*. 2014. ISBN: 9780081024355.

[27] Greffe, T., Smith, R., Sherman, M., Harrison, F., Earnshaw, H., Grefenstette, B., Hennessy, J. and Nikzad, S. Characterization of Low Light Performance of a CMOS sensor for Ultraviolet Astronomical Applications. (Dec. 2021), p. 13. DOI: 10.48550/arxiv.2112.01691. arXiv: 2112.01691. URL: https://arxiv.org/abs/2112.01691v3.

[28] Yahiaoui, L., Horgan, J., Deegan, B., Yogamani, S., Hughes, C. and Denny, P. Overview and empirical analysis of ISP parameter tuning for visual perception in autonomous driving. *Journal of Imaging* 5.10 (2019). ISSN: 2313433X. DOI: 10.3390/jimaging5100078.

[29] Hartley, R. and Zisserman, A. *Multiple View Geometry in Computer Vision*. Cambridge University Press, Mar. 2004. ISBN: 9780521540513.

[30] Zhang, Z. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.11 (2000), pp. 1330–1334. ISSN: 01628828. DOI: 10.1109/34.888718.

[31] Lumenera and Teledyne. *The most important camera parameters for aerial mapping*. 2018. URL: https://www.lumenera.com/resources/case-studies.html (visited on 09/27/2022).

[32] Zhang, Z., Rebecq, H., Forster, C. and Scaramuzza, D. Benefit of large field-of-view cameras for visual odometry. *Proceedings - IEEE International Conference on Robotics and Automation* 2016-June (2016), pp. 801–808. ISSN: 10504729. DOI: 10.1109/ICRA.2016.7487210.

[33] Handa, A., Newcombe, R. A., Angeli, A. and Davison, A. J. Real-time camera tracking: When is high frame-rate best?: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 7578 LNCS.PART 7 (2012), pp. 222–235. ISSN: 03029743. DOI: 10.1007/978-3-642-33786-4_17.

[34] Zhang, K., Xie, J., Snavely, N. and Chen, Q. Depth sensing beyond lidar range. *Proceedings of the IEEE Computer Society Conference on Computer Vision and*

Pattern Recognition (2020), pp. 1689–1697. ISSN: 10636919. DOI: 10.1109/CVPR42600.2020.00176. arXiv: 2004.03048.

[35] Titterton, D. H. and Weston, J. L. *Strapdown Inertial Navigation Technology*. 2004, p. 588. ISBN: 0863413587.

[36] Ribeiro, N. F. and Santos, C. P. Inertial measurement units: A brief state of the art on gait analysis. *ENBENG 2017 - 5th Portuguese Meeting on Bioengineering, Proceedings* January (2017). DOI: 10.1109/ENBENG.2017.7889458.

[37] Collin, J., Davidson, P., Kirkko-Jaakkola, M. and Leppäkoski, H. Inertial sensors and their applications. *Handbook of Signal Processing Systems* (Oct. 2018), pp. 51–85. DOI: 10.1007/978-3-319-91734-4_2.

[38] Conlin, W. T. Review Paper: Inertial Measurement. May (2017), pp. 1–10. arXiv: 1708.04325. URL: http://arxiv.org/abs/1708.04325.

[39] Deppe, O., Dorner, G., König, S., Martin, T., Voigt, S. and Zimmermann, S. MEMS and FOG technologies for tactical and navigation grade inertial sensors—Recent improvements and comparison. *Sensors (Switzerland)* 17.3 (2017). ISSN: 14248220. DOI: 10.3390/s17030567.

[40] Parkinson, B. W., Spilker, J. J., Axelrad, P., Enge, P. and American Institute of Aeronautics and Astronautics. *The Global Positioning System : Theory and Applications, Volume I*. American Institute of Aeronautics and Astronautics, 1996, p. 781. ISBN: 1600864198.

[41] Hein, G. W. Status, perspectives and trends of satellite navigation. *Satellite Navigation* 1.1 (Dec. 2020), pp. 1–12. ISSN: 26621363. DOI: 10.1186/S43020-020-00023-X.

[42] Parkinson, B. W., Spilker, J. J., Axelrad, P. and Enge, P. *Global Positioning System : theory and applications. Volume II*. American Institute of Aeronautics and Astronautics, Inc, 1996, p. 665. ISBN: 1-60086-639-5.

[43] Gurbuz, G., Gormus, K. S. and Altan, U. Investigation of the Air Quality Change Effect on GNSS Signals. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 4.4W4 (2017), pp. 229–233. ISSN: 21949050. DOI: 10.5194/isprs-annals-IV-4-W4-229-2017.

[44] Van Diggelen, F. S. T. *A-GPS : Assisted GPS, GNSS, and SBAS*. Artech House, 2009, p. 380. ISBN: 9781596933750.

[45] Li, X., Wang, B., Li, X., Huang, J., Lyu, H. and Han, X. Principle and performance of multi-frequency and multi-GNSS PPP-RTK. *Satellite Navigation* 3.1 (Dec. 2022), pp. 1–11. ISSN: 26621363. DOI: 10.1186/S43020-022-00068-0.

[46] Zhang, H., Aldana-Jague, E., Clapuyt, F., Wilken, F., Vanacker, V. and Van Oost, K. Evaluating the potential of post-processing kinematic (PPK) georeferencing for UAV-based structure-from-motion (SfM) photogrammetry and surface change detection. *Earth Surface Dynamics* 7.3 (Sept. 2019), pp. 807–827. ISSN: 2196632X. DOI: 10.5194/ESURF-7-807-2019.

[47] Weber, G., Dettmering, D. and Gebhard, H. Networked transport of RTCM via internet protocol (NTRIP). *International Association of Geodesy Symposia* 128 (2005), pp. 60–64. ISSN: 21979359. DOI: 10.1007/3-540-27432-4_11.

[48] Rehder, J., Nikolic, J., Schneider, T., Hinzmann, T. and Siegwart, R. Extending kalibr: Calibrating the extrinsics of multiple IMUs and of individual axes. *Proceedings - IEEE International Conference on Robotics and Automation* 2016-June (2016), pp. 4304–4311. ISSN: 10504729. DOI: 10.1109/ICRA.2016.7487628.

[49] Furgale, P., Rehder, J. and Siegwart, R. Unified temporal and spatial calibration for multi-sensor systems. *IEEE International Conference on Intelligent Robots and Systems* (2013), pp. 1280–1286. ISSN: 21530858. DOI: 10.1109/IROS.2013.6696514.

[50] Szeliski, R. *Computer Vision*. Texts in Computer Science. London: Springer London, 2011. ISBN: 978-1-84882-934-3. DOI: 10.1007/978-1-84882-935-0.

[51] Mohammadmoradi, H., Gnawali, O. and Szalay, A. Accurately initializing real time clocks to provide synchronized time in sensor networks. *2017 International Conference on Computing, Networking and Communications, ICNC 2017* (2017), pp. 486–490. DOI: 10.1109/ICCNC.2017.7876177.

[52] Behrendt, K., Fodero, K., Laboratories, S. E. et al. The perfect time: An examination of time-synchronization techniques. (2006). URL: researchgate.net/publication/266099236.

[53] Rupert, M. *Linux Kernel Support for IEEE 1588 Hardware Timestamping*. 2021. URL: https://www.renesas.com/eu/en/document/whp/linux-kernel-support-ieee-1588-hardware-timestamping (visited on 09/27/2022).

[54] Ogrizovic, V. R., Marendic, J., Renovica, S., Delcev, S. and Gucevic, J. Testing GPS generated 1PPS against a rubidium standard. *Acta Imeko* 2.1 (2013), p. 7. DOI: 10.21014/acta_imeko.v2i1.47.

[55] Refan, M. H. and Valizadeh, H. Computer network time synchronization using a Low Cost GPS engine. *Iranian Journal of Electrical and Electronic Engineering* 8.3 (2012), pp. 206–216. ISSN: 17352827.

[56] FLIR-Teledyne. *FLIR Blackfly S BFS-U3-16S2 Datasheet*. URL: http://softwareservices.flir.com/BFS-U3-16S2/latest/Model/spec.html (visited on 09/26/2022).

[57] Theia Technologies. *SY110 / MY110 Specification*. URL: https://thttheia-live-558c38c6e3f64c7299bc1b37e-6693e9d.divio-media.org/documents/theia-SY110-lens-specification.pdf (visited on 09/26/2022).

[58] OmniVision. *OmniVision OV9281 Sensor Datasheet*. URL: https://www.ovt.com/sensors/OV9281 (visited on 09/26/2022).

[59] VECTORNAV. *VECTORNAV*. URL: https://www.vectornav.com/ (visited on 09/26/2022).

[60] IXBlue. *iXBlue*. URL: https://www.ixblue.com/photonics-space/inertial-navigation-for-space/ (visited on 09/26/2022).

[61] U-blox. *GPS-based timing: considerations with u-blox 6 GPS receivers*. URL: https://www.u-blox.com (visited on 09/26/2022).

[62] Murata Electronics. *Murata SCHA6XX Datasheet*. URL: https://www.murata.com/en-global/products/sensor/gyro/overview/lineup/scha600 (visited on 09/26/2022).

[63] U-blox. *ZED-F9R Datasheet*. URL: https://www.u-blox.com/en/product/zed-f9r-module (visited on 09/26/2022).

[64] Swift Navigation. *Swift Navigation*. URL: https://www.swiftnav.com/ (visited on 09/26/2022).

[65] Serpentrio. *Serpentrio*. URL: https://www.septentrio.com/en (visited on 09/26/2022).

[66] Leica. *Leica Geosystems*. URL: https://leica-geosystems.com/ (visited on 09/26/2022).

[67] U-blox. *ANN-MB GNSS Antenna Datasheet*. URL: https://www.u-blox.com (visited on 09/26/2022).

[68] FLIR-Teledyne. *FLIR Blackfly S BFS-U3-16S2 Technical Reference*. URL: http://softwareservices.flir.com/BFS-U3-16S2/latest/Model/spec.html (visited on 09/26/2022).

[69] EMVA. *GenICam Standard Generic Interface for Cameras*. URL: https://www.emva.org/standards-technology/genicam/ (visited on 08/15/2022).

[70] FLIR-Teledyne. *Spinnaker SDK | Teledyne FLIR*. URL: https://www.flir.com/products/spinnaker-sdk/ (visited on 09/26/2022).

[71] FLIR-Teledyne. *FLIR Blackfly S EMVA 1288 Imaging Performance*. URL: http://softwareservices.flir.com/BFS-U3-16S2/latest/EMVA/EMVA-Local.html (visited on 09/26/2022).

[72] FLIR-Teledyne. *Spinnaker C++: Programmer's Guide*. URL: http://softwareservices.flir.com/Spinnaker/latest/%7B%5C_%7Dprogrammer%7B%5C_%7Dguide.html (visited on 09/26/2022).

[73] Linux Foundation. *Video for Linux API — The Linux Kernel documentation*. URL: https://www.kernel.org/doc/html/v4.17/media/uapi/v4l/v4l2.html (visited on 09/26/2022).

[74] NVIDIA Corporation. *NVIDIA Jetson Xavier NX Developer Kit Carrier Board P3509_A01 Specifications*. 2020. URL: https://developer.nvidia.com/embedded/downloads (visited on 09/26/2022).

[75] NVIDIA Corporation. *NVIDIA Jetson Xavier NX Series System-on-Module Datasheet*. 2021. URL: https://developer.nvidia.com/embedded/downloads (visited on 09/26/2022).

[76] MIPI Alliance. *Driving the Wires of Automotive*. 2019. URL: https://www.mipi.org/driving-the-wires-of-automotive (visited on 09/26/2022).

[77] NVIDIA Corporation. *Accelerated Gstreamer User Guide*. URL: https://developer.nvidia.com/embedded/downloads (visited on 09/26/2022).

[78] Murata Electronics. *SCHA600 Inertial Measurement Unit*. URL: https://www.murata.com/en-eu/products/sensor/gyro/overview/lineup/scha600 (visited on 09/26/2022).

[79] White, E. *Making Embedded Systems*. 2011, p. 330. ISBN: 9781449302146.

[80] Arduino. *Arduino Documentation*. URL: https://docs.arduino.cc/ (visited on 09/27/2022).

[81] Murata Electronics. *Murata SCHA6XX Carrier PCB*. 2022. URL: https://www.murata.com/en-global/products/sensor/gyro/overview/lineup/scha600 (visited on 09/26/2022).

[82] Arduino. *Arduino Mega Documentation*. URL: https://docs.arduino.cc/hardware/mega-2560 (visited on 09/26/2022).

[83] Arduino. *Arduino Due Documentation*. URL: https://docs.arduino.cc/hardware/due (visited on 09/26/2022).

[84] Arduino. *Arduino Ethernet Shield Documentation*. URL: https://docs.arduino.cc/retired/shields/arduino-ethernet-shield-without-poe-module (visited on 09/26/2022).

[85] Bogatin, E. *Signal and Power Integrity Simplified*. Pearson, 2003. ISBN: 9780130669469.

[86] Wiznet Corporation. *Wiznet W5500 Datasheet*. 2013. URL: https://www.wiznet.io/product-item/w5500/ (visited on 09/26/2022).

[87] Silberschatz, A., Galvin, P. B. and Gagne, G. *Operating System Concepts*. Wiley, 2018. ISBN: 978-1-118-06333-0.

[88] Raspberry Foundation. *Raspberry Pi Documentation - RP2040*. URL: https://www.raspberrypi.com/documentation/microcontrollers/rp2040.html (visited on 09/28/2022).

[89] Wiznet Corporation. *W5500-EVB-Pico*. URL: https://www.wiznet.io/product-item/w5500-evb-pico/ (visited on 09/26/2022).

[90] Wu, T. L., Buesink, F. and Canavero, F. Overview of signal integrity and EMC design technologies on PCB: Fundamentals and latest progress. *IEEE Transactions on Electromagnetic Compatibility* 55.4 (2013), pp. 624–638. ISSN: 00189375. DOI: 10.1109/TEMC.2013.2257796.

[91] Murata Electronics. *Murata SCHA6XX Assembly Guide*. 2022. URL: https://www.murata.com/en-global/products/sensor/gyro/overview/lineup/scha600 (visited on 09/26/2022).

[92] U-blox. *ZED-F9P Application board*. URL: https://www.u-blox.com/en/product/zed-f9p-module (visited on 09/26/2022).

[93] U-blox. *ZED-F9R Application board*. URL: https://www.u-blox.com/en/product/zed-f9r-module (visited on 09/26/2022).

[94] U-blox. *ZED-F9P Datasheet*. URL: https://www.u-blox.com/en/product/zed-f9p-module (visited on 09/26/2022).

[95] U-blox. *u-center GNSS evaluation software*. URL: https://www.u-blox.com/en/product/u-center (visited on 09/26/2022).

[96] U-blox. *ZED-F9R Integration Manual*. URL: https://www.u-blox.com/en/product/zed-f9r-module (visited on 09/26/2022).

[97] U-blox. *ZED-F9 HPS 1.32 Interface description*. URL: https://www.u-blox.com (visited on 09/26/2022).

[98] RTKLIB. *RTKLIB: An Open Source Program Package for GNSS Positioning*. URL: https://www.rtklib.com/ (visited on 09/26/2022).

[99] U-blox. *Achieving Centimeter Level Performance with Low Cost Antennas*. URL: https://www.u-blox.com/en/publication/white-paper/achieving-centimeter-level-performance-low-cost-antennas (visited on 09/26/2022).

[100] U-blox. *GNSS antennas : RF Design Considerations for u-blox GNSS Receivers*. URL: https://www.u-blox.com (visited on 09/26/2022).

[101] Intel Corporation. *USB 3.0 Radio Frequency Interference Impact on 2.4 GHz Wireless Devices*. 2012. URL: https://www.usb.org/sites/default/files/327216.pdf (visited on 09/26/2022).

[102] FLIR-Teledyne. *FLIR Blackfly S Installation Guide*. URL: http://softwareservices.flir.com/BFS-U3-16S2/latest/Model/spec.html (visited on 09/26/2022).

[103] Markforged. *Onyx - Composite 3D Printing Material*. URL: https://markforged.com/materials/plastics/onyx (visited on 09/26/2022).

[104] Prusa Research. *Prusa 3D printers*. URL: https://www.prusa3d.com/ (visited on 09/26/2022).

[105] NVIDIA Corporation. *NVIDIA Jetson Xavier NX System-on-Module Thermal Design Guide*. 2020. URL: https://developer.nvidia.com/embedded/downloads (visited on 09/26/2022).

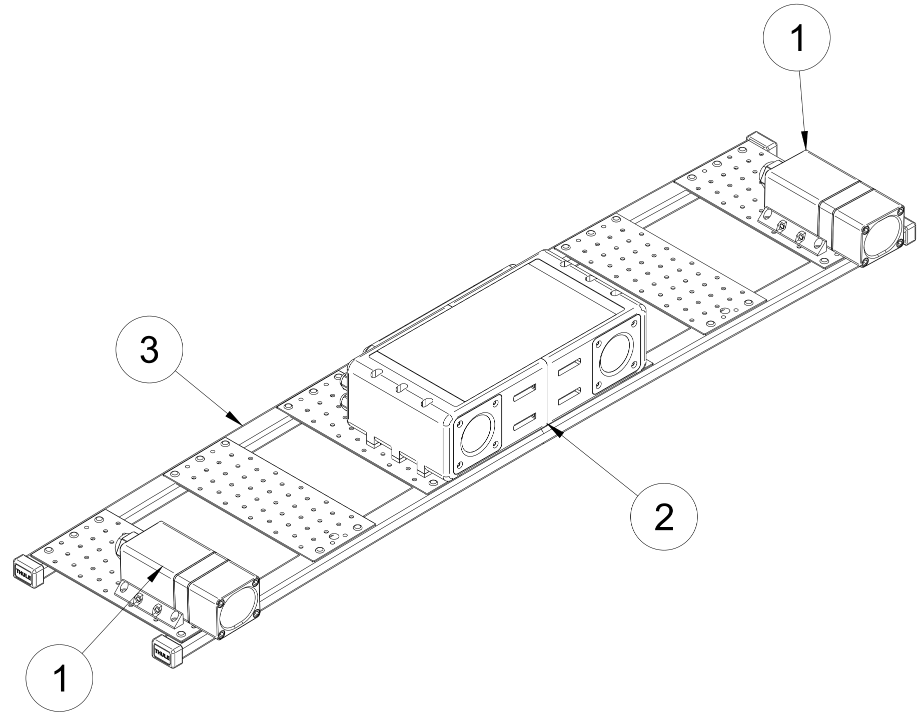
[106] Moran, M. J., Shapiro, H. N., Boettner, D. D. and Bailey, M. B. *Fundamentals of Engineering Thermodynamics*. Wiley, 2014. ISBN: 1118820444.

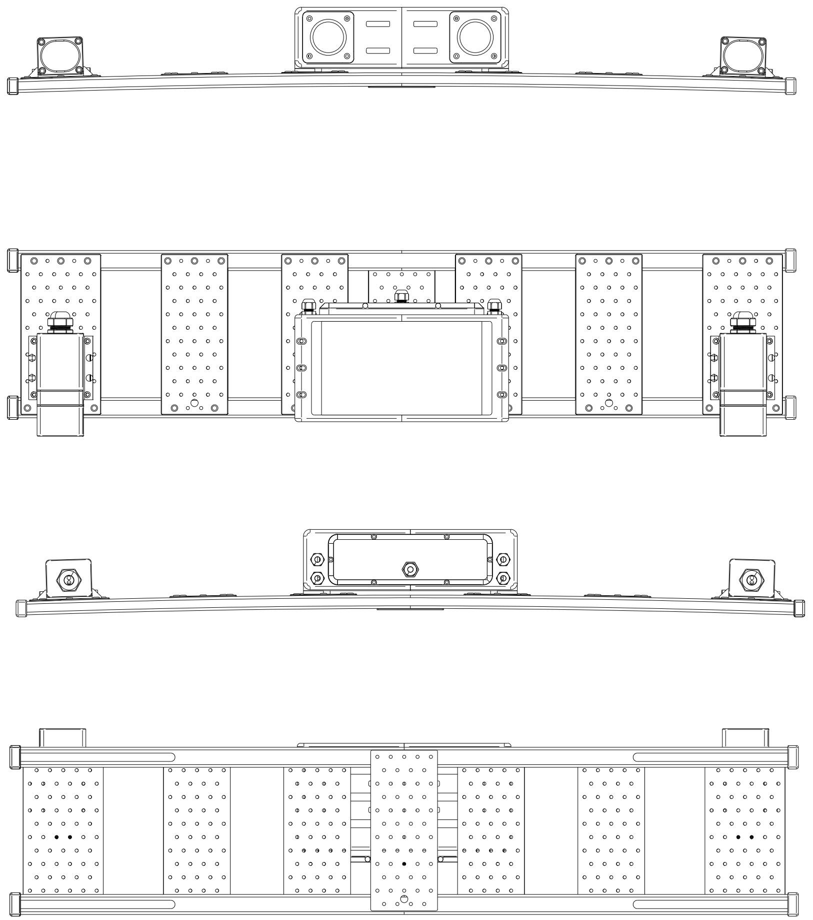
APPENDIX A: TECHNICAL DRAWINGS

Drawings for the main parts of the implemented system. An index page is presented followed by drawings for the machine vision cameras, integrated vision system and mounting system. Lastly, drawings for the timing module's versions 1.1 and 1.2 are presented and the spatial relation of the timing module with respect to the integrated vision system's chassis.

Parts List

Item	Qty	Part Number	Description
1	2	MachineVisionCamera	Mounting solution for icecube–formfactor machinevision camera
2	1	IntegratedCameraSystem	Mounting system for shielded stereocamera, RTK-GNSS module and computation unit
3	1	ThuleRoofrack	Thule SquareBar roofrack combined with tieplates for mounting sensor hardware

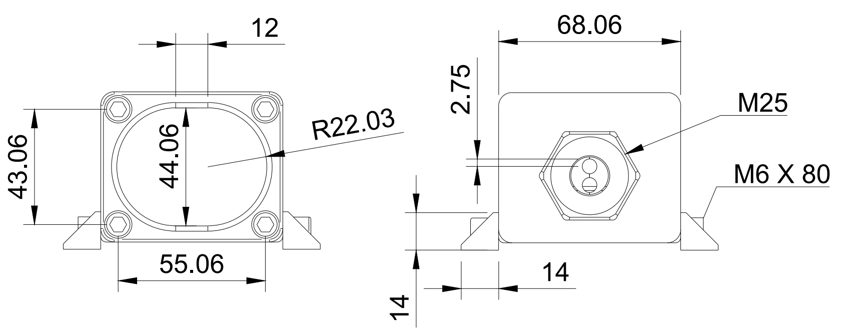


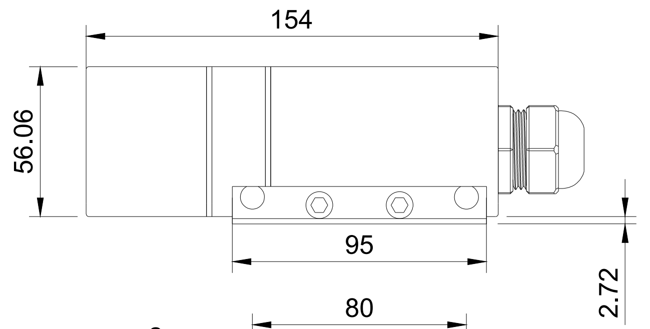


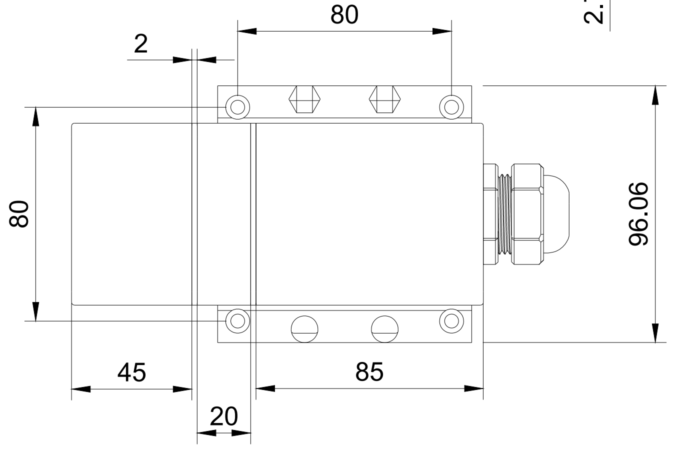
Dept.	Technical reference	Created by Christian Kaarre	Approved by
		Document type Tech. Drawing	Document status First issue
		Title Camera System	DWG No.
		Index	Rev. 00 · Date of issue Jun 2022 · Sheet 1/7

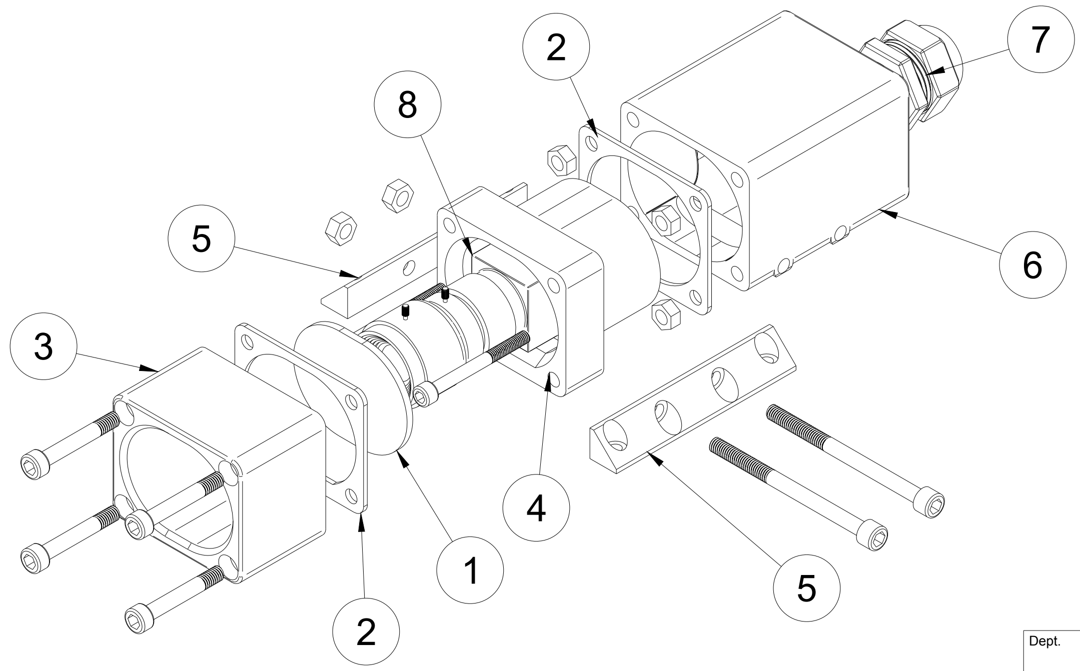
Parts List

Item	Qty	Part Number	Description
1	1	EnclosureShield	Protective shield for optics, material e.g. acrylic/polycarbonate
2	1	EnclosureGaskets	Interface gaskets for waterproofing between parts, e.g. NBR/silicone
3	1	LensShieldCase	Front part housing the shield, PLA
4	1	CamMountingCase	Mount for camera and lens shielding, PLA 75% infill
5	1	EnclosureRails	Rails for mounting camera housing into roofracks
6	1	PassthroughCase	Interface for fastening camera into place including passthrough for cables, PLA
7	1	EnclosureSocketseal	M25 socketseal for cable passthrough
8	1	IcecubeModel	FLIR BlackFly S + Theia Technologies SY110M







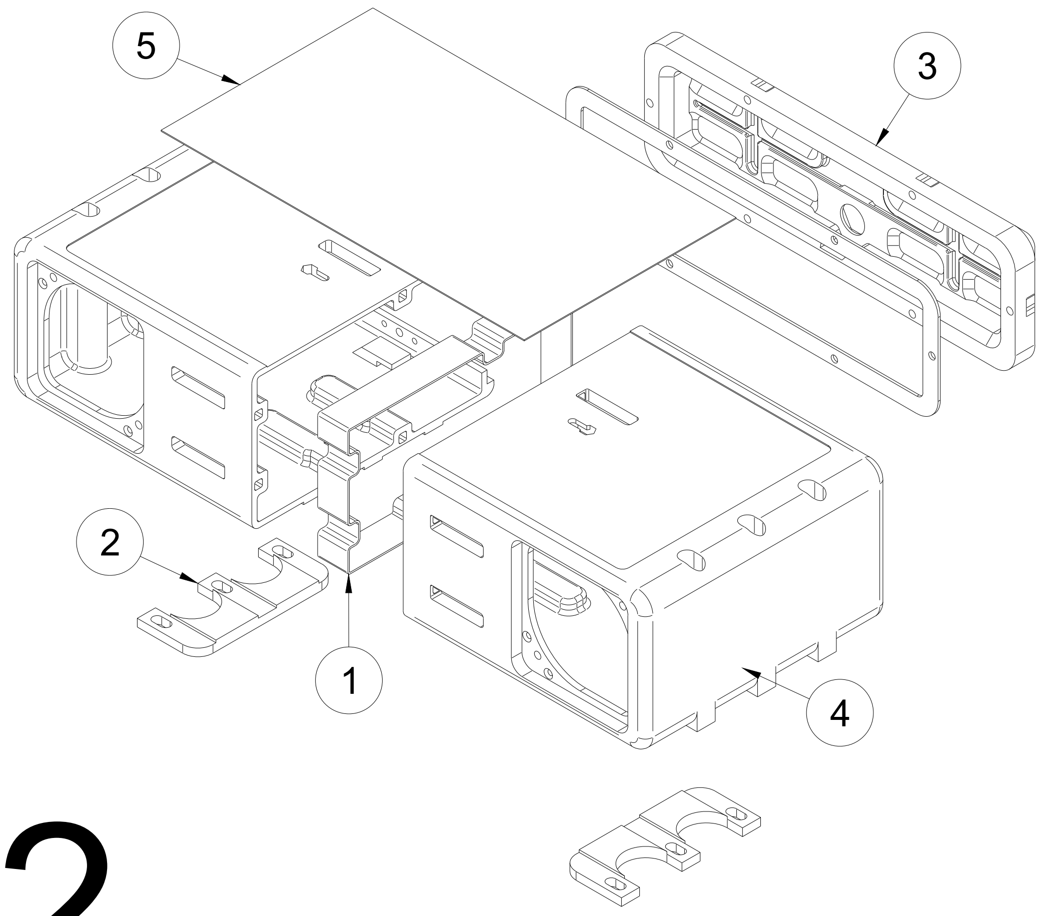


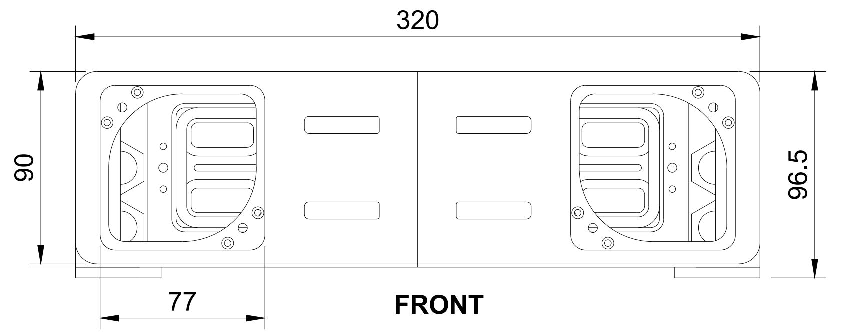
1

Dept.	Technical reference	Created by	Approved by
		Christian Kaarre	
		Document type: Tech. Drawing	Document status: First issue
		Title: Camera System MachineVision	DWG No.
			Rev. 00 / Date of issue Jun 2022 / Sheet 2/7

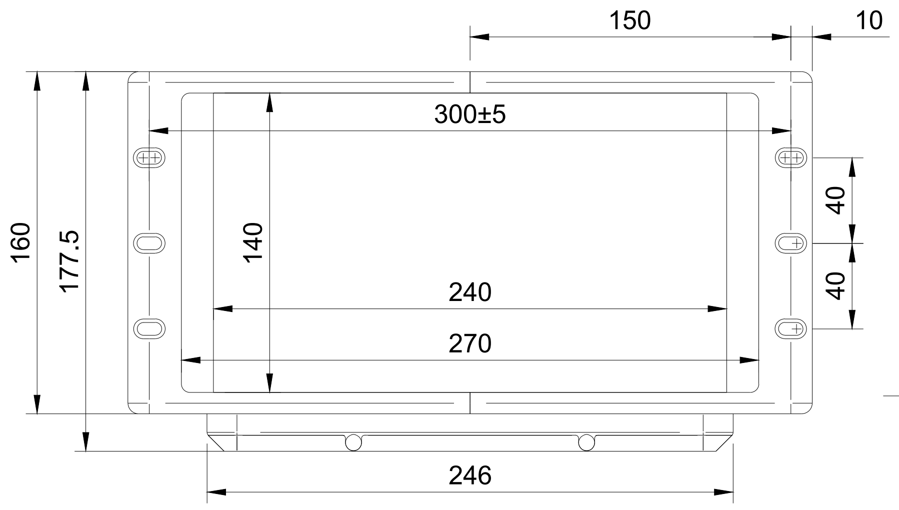
Parts List

Item	Qty	Part Number	Description
1	1	ChassisReinforcement	Reinforcement on the intersection of the frame/chassis pieces
2	1	ChassisSpacer	Spacers for clearance with the roofrack's rivets
3	1	Battery	Battery housing for 4 x 18650 cells with 2 x Keystone 1048P holders
4	1	ChassisFrame	Frame consisting of mirrored sides, connecting with five screwpoints and battery housing
5	1	GroundPlane	Steel ground plane for uBlox ANN-MB patch antenna, mounting to chassis with screws

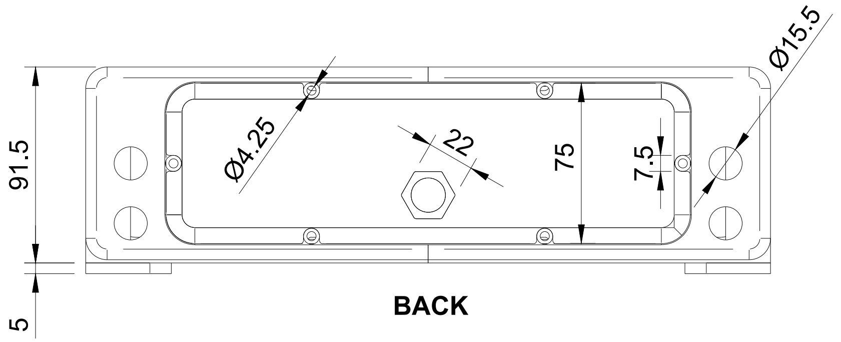




FRONT



TOP

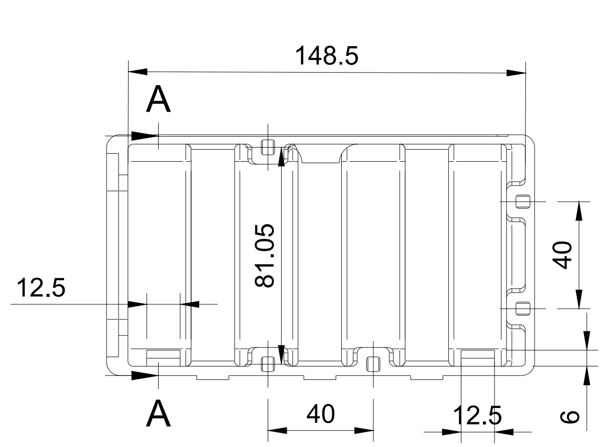


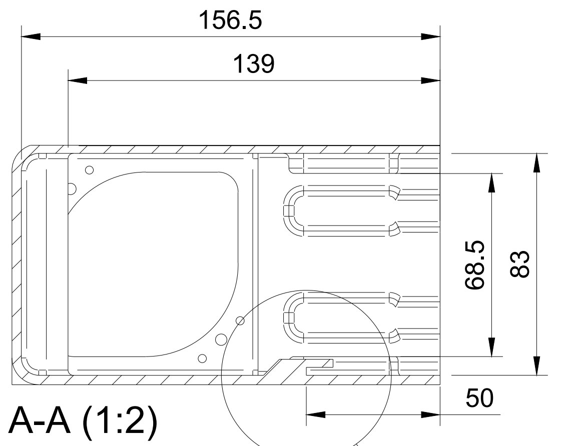
BACK

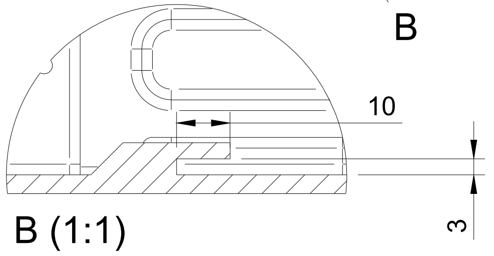
2

Dept.	Technical reference	Created by	Approved by
		Christian Kaarre	
		Document type: Tech. Drawing	Document status: First issue
		Title: Camera System IntegratedCamera	DWG No.: 1/4
			Rev.: 00 / Date of issue: Jun 2022 / Sheet: 3/7

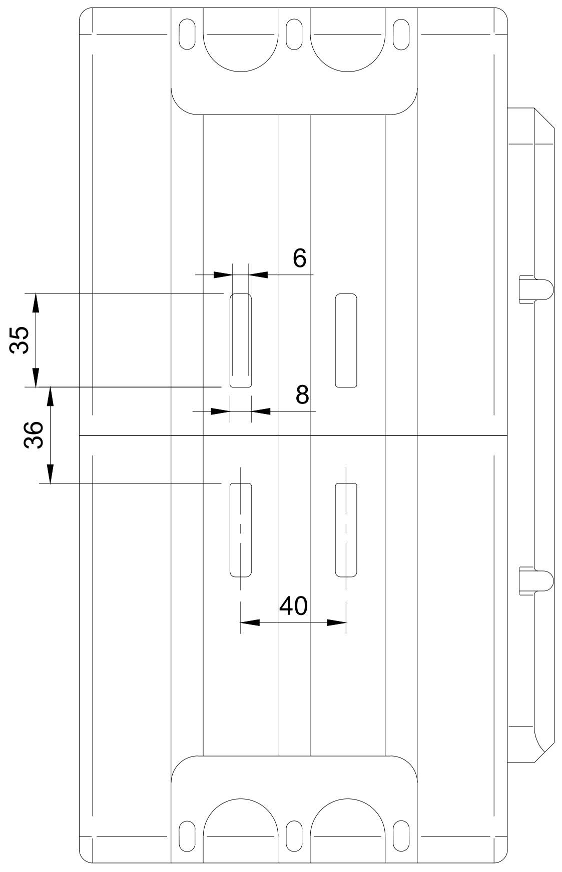
DEVICE CORE SUPPORT/LOCKING





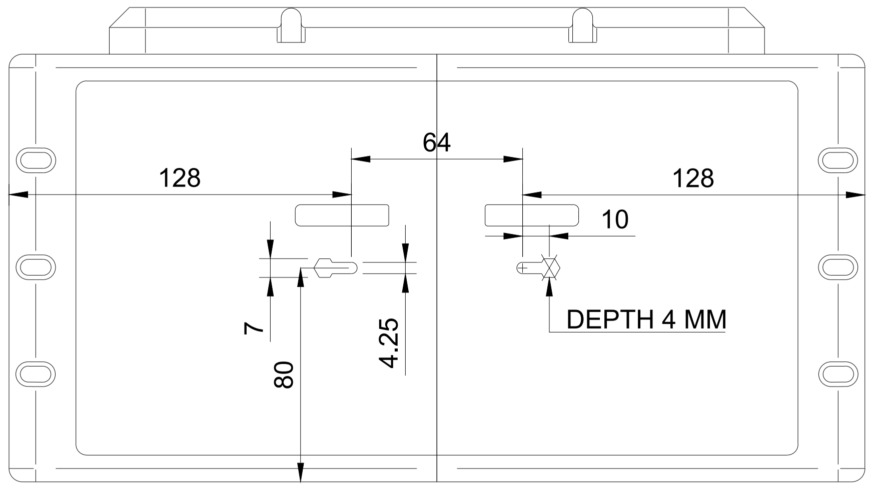


CONNECTING CHASSIS SIDES



5 MOUNTING POINTS
M4 X 40MM INT. HEX

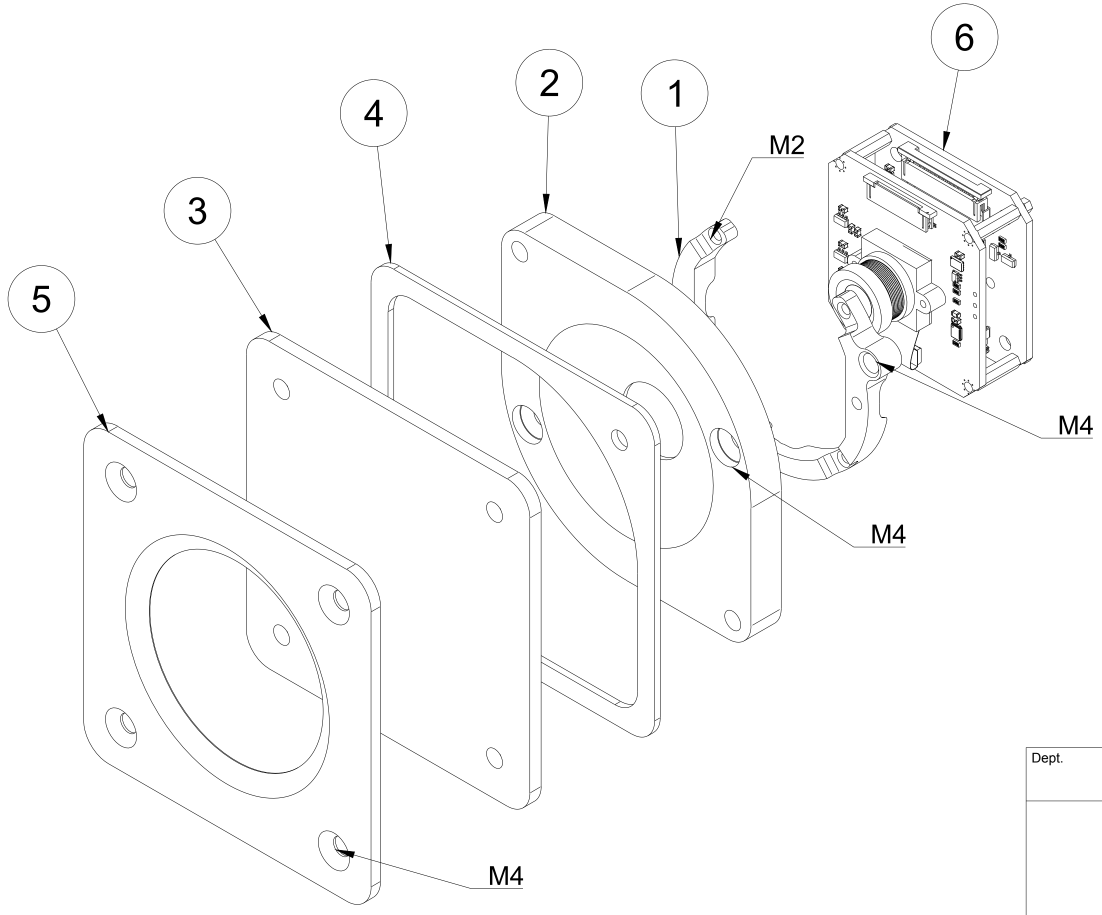
ANTENNA MOUNTING POINTS

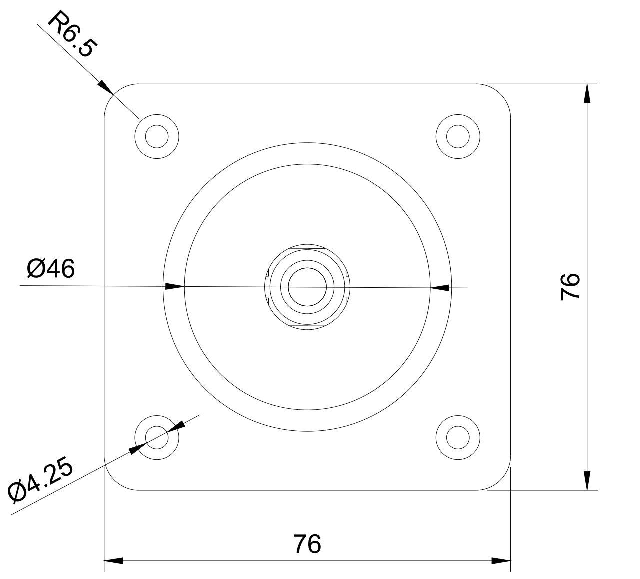


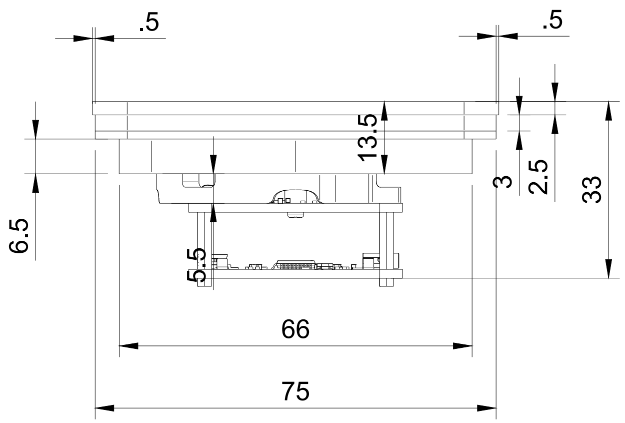
Dept.	Technical reference	Created by	Approved by
		Christian Kaarre	
		Document type: Tech. Drawing	Document status: First issue
		Title: Camera System IntegratedCamera	DWG No. 2/4
			Rev. 00 / Date of issue Jun 2022 / Sheet 4/7

Parts List

Item	Qty	Part Number	Description
1	1	CameraInterface	Interface bracket connecting camera to lens hood, Onyx
2	1	CameraShieldInterface	Lens hood securing shield into camera structure (two points), PLA
3	1	CameraShield	Shield connecting to lens hood (two points) and chassis (two points), acrylic/polycarbonate
4	1	CameraGasket	Gasket for sealing shield to chassis
5	1	CameraFrontPlate	Front plate for screw heads, secured to chassis, passthroughs for camerastructure, PLA
6	1	Camera	Arducam RAW/OV9281 Jetson Module



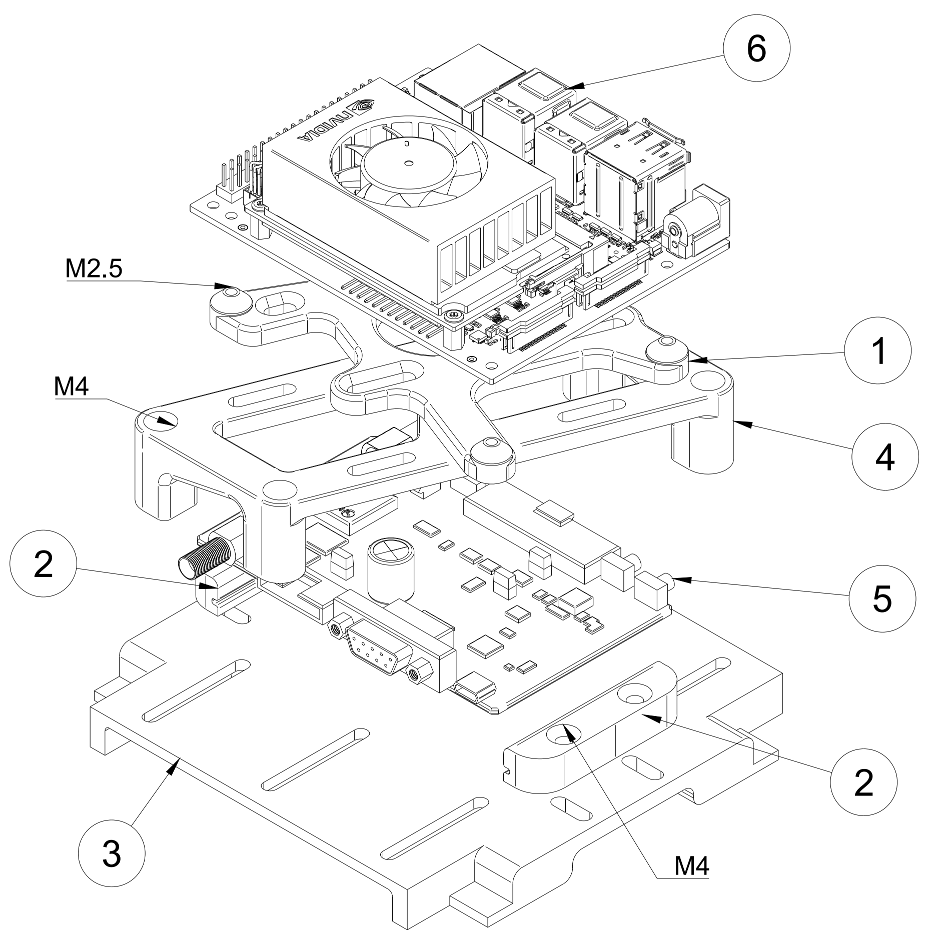


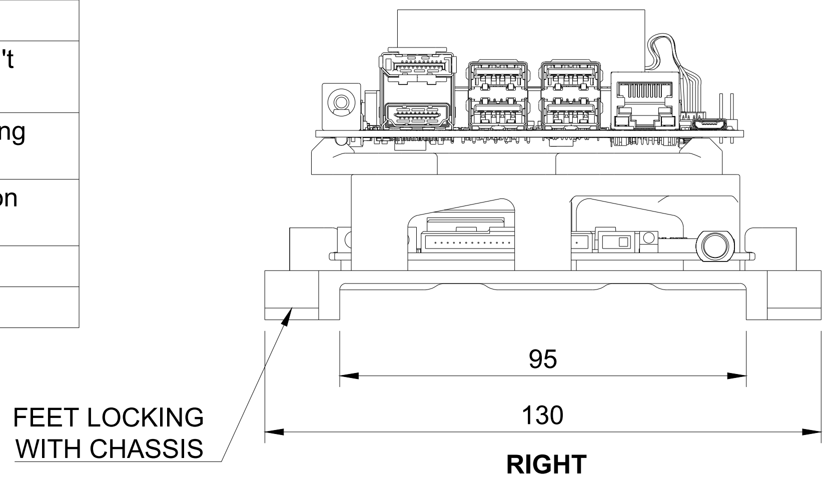


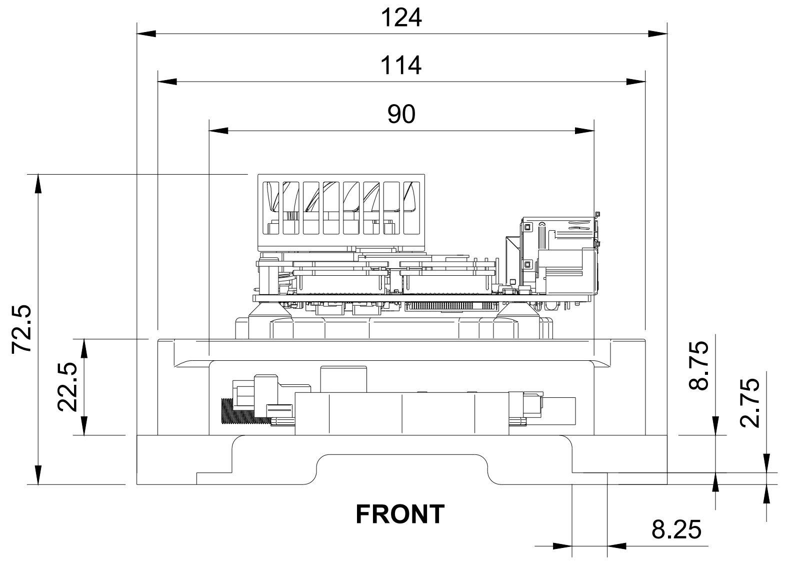
Dept.	Technical reference	Created by	Approved by
		Christian Kaarre	
		Document type: Tech. Drawing	Document status: First issue
		Title: Camera System IntegratedCamera	DWG No.: 3/4
			Rev.: 00 / Date of issue: Jun 2022 / Sheet: 5/7

Parts List

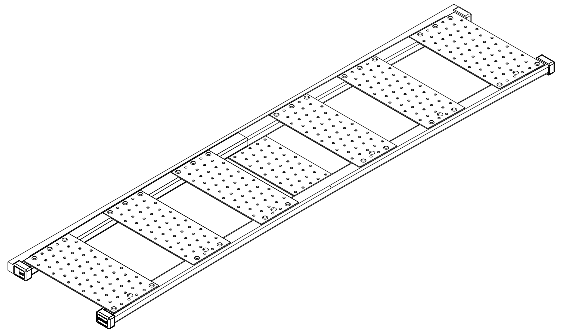
Item	Qty	Part Number	Description
1	1	MountXavier	Bracket for mounting Jetson to rails on spacer
2	1	MountUBlox	Mountpoints for uBlox development board, board doesn't have drilled mounting points
3	1	MountBase	Base mounting, fixes structure to chassis with interlocking mechanism
4	1	MountSpacer	Spacer for stacking development kits, connects to Jetson bracket and base via screws on rails
5	1	UBlox	UBlox F9R development kit
6	1	Xavier	NVidia Jetson Xavier NX development kit

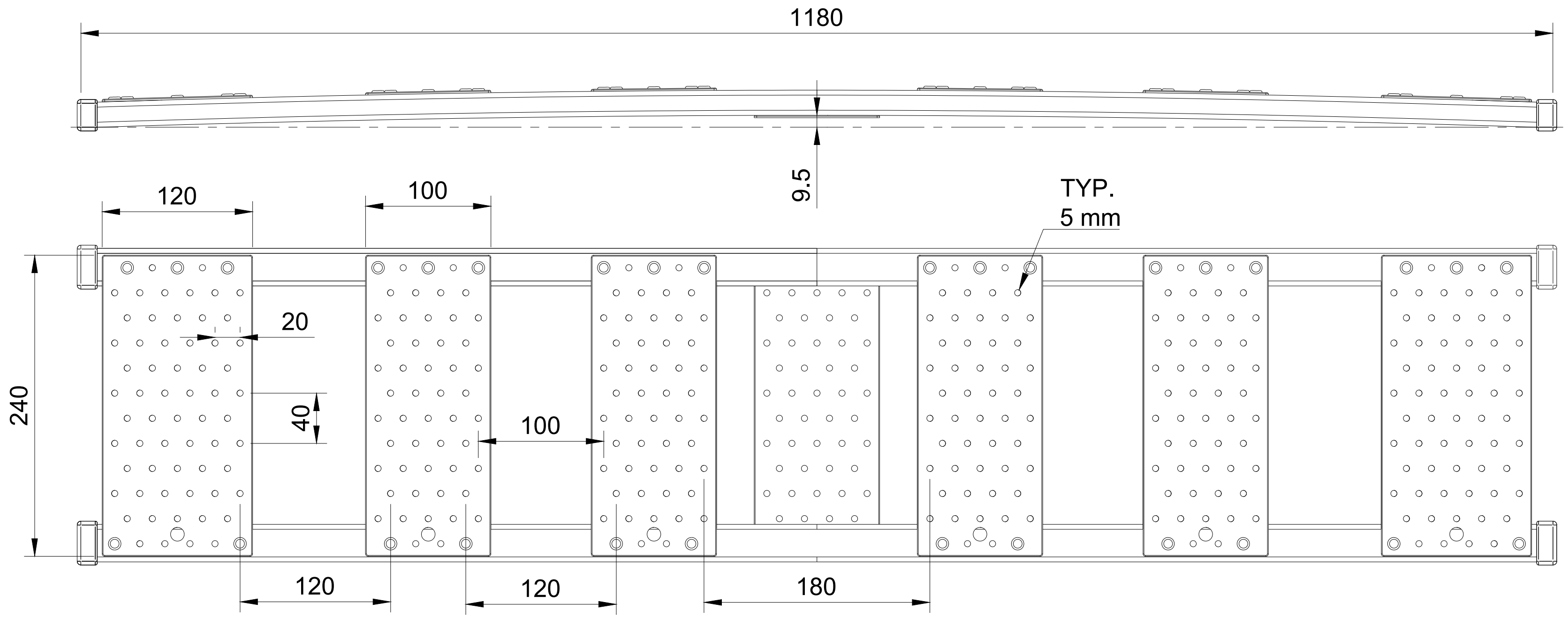






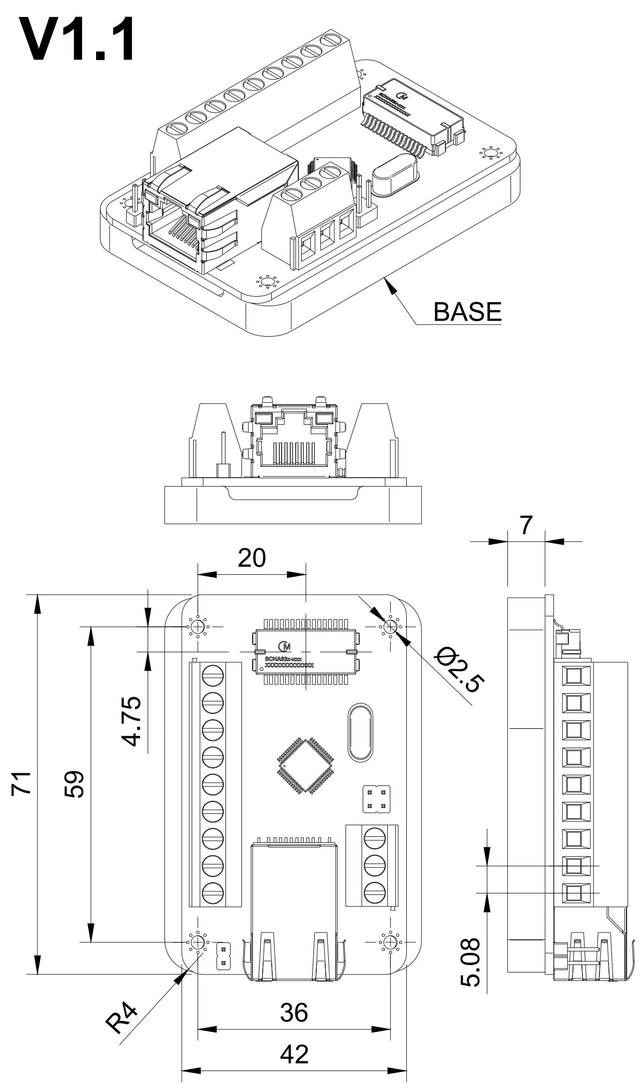
Dept.	Technical reference	Created by	Approved by
		Christian Kaarre	
		Document type: Tech. Drawing	Document status: First issue
		Title: Camera System IntegratedCamera	DWG No.: 4/4
			Rev.: 00 / Date of issue: Jun 2022 / Sheet: 6/7

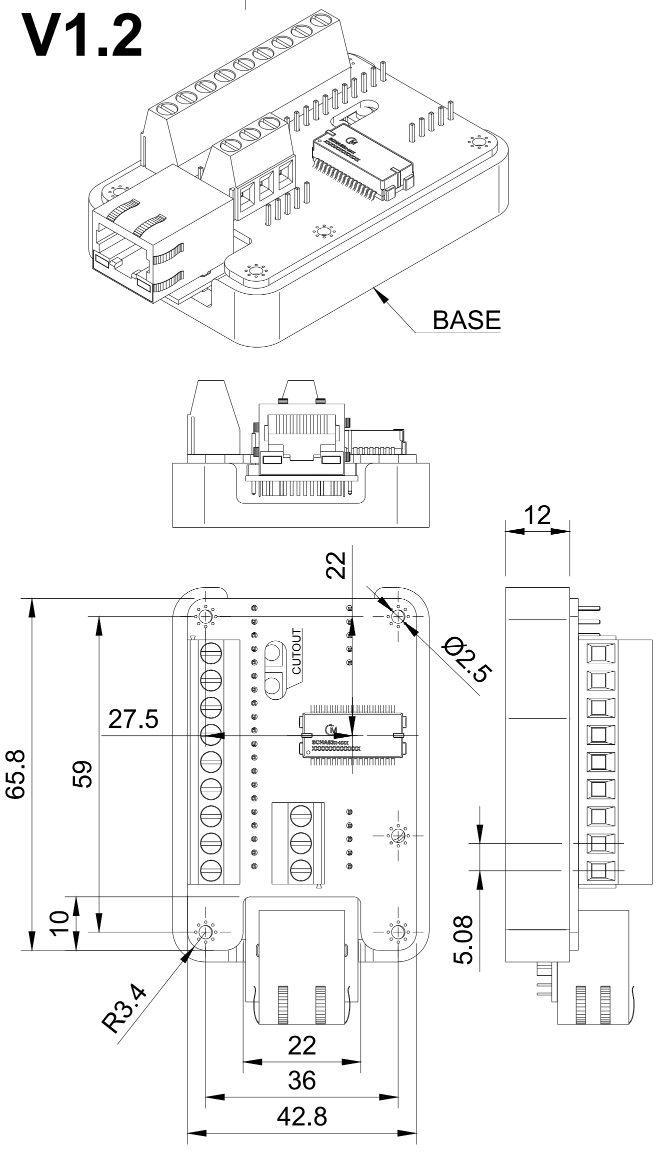


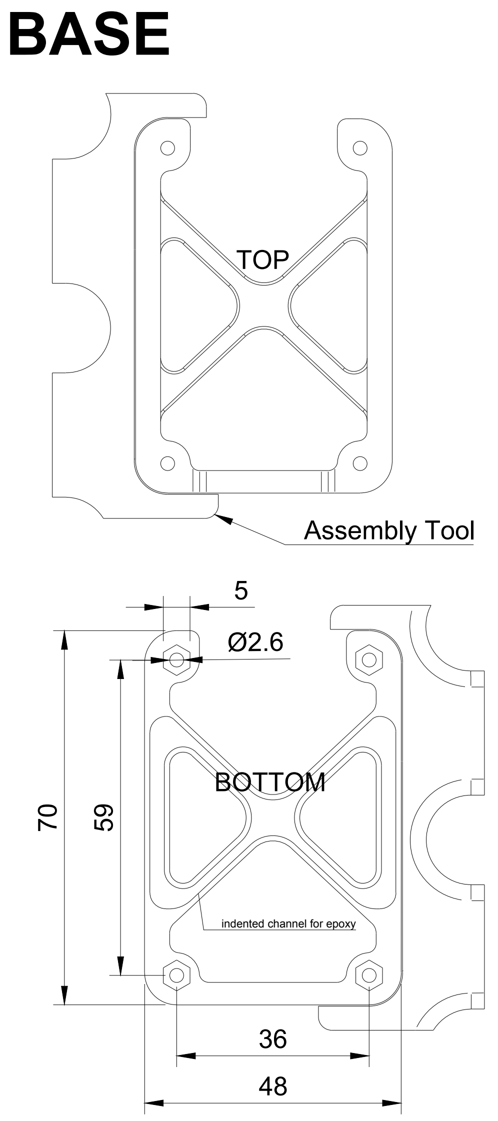


3

Dept.	Technical reference	Created by Christian Kaarre	Approved by
		Document type Tech. Drawing	Document status First issue
		Title Camera System Thule Roofrack	DWG No.
			Rev. 00 · Date of issue Jun 2022 · Sheet 7/7





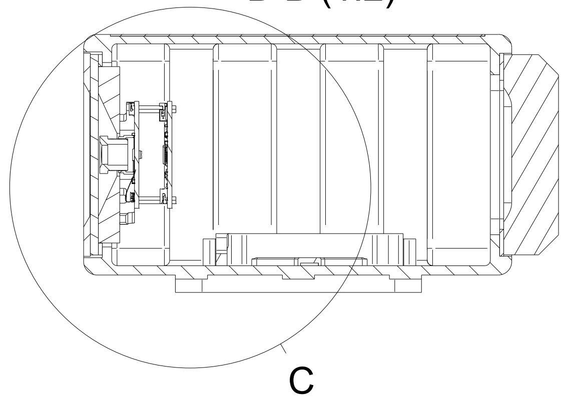


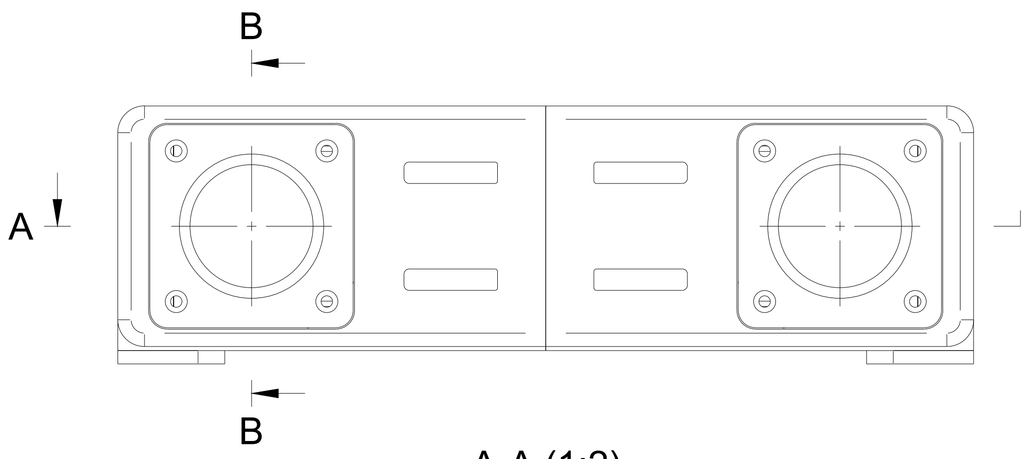
NOTES:

- * some SMD components and most of the vias are excluded from these drawings, check design files for full layout and BOM
- * the mounting base part for both versions is the same and can either be integrated into the full system's assembly or epoxied into place
  - ** the AssemblyTool part in BASE is not intended for operational use, only for pcb alignment if pcb is attached separately

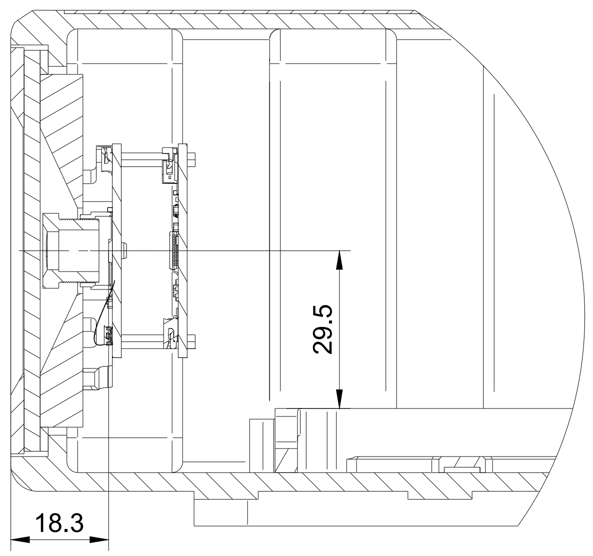
Dept.	Technical reference	Created by Christian Kaarre	Approved by
		Document type Tech. Drawing	Document status First Issue
		Title Camera System IMUMount	DWG No.
			Rev. 00 · Date of issue Jun 2022 · Sheet 1/2

B-B (1:2)

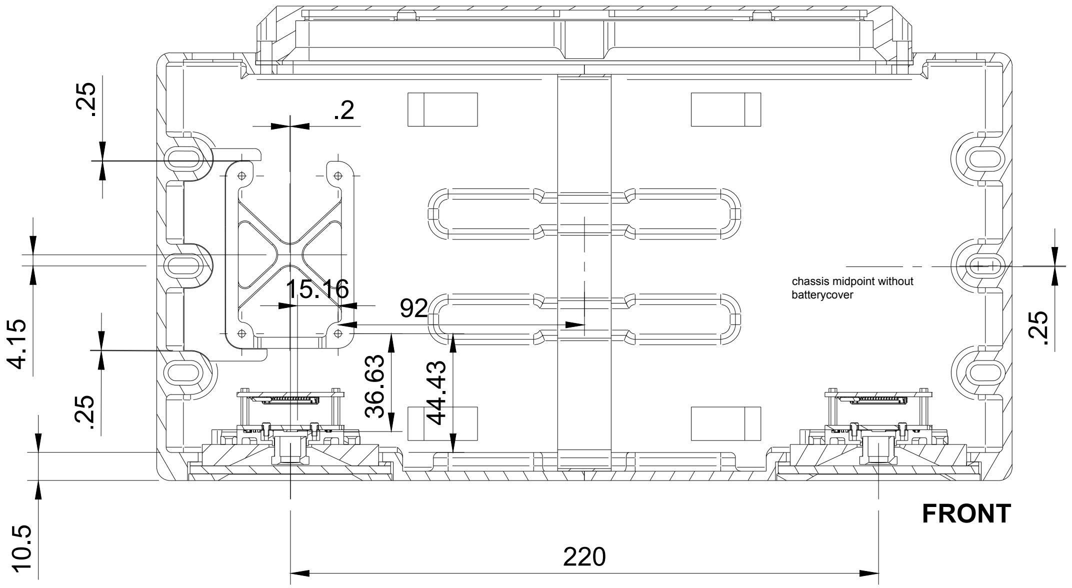




C (1:1)



A-A (1:2)



NOTES:

- * version 1.2 of the IMUMount is used in this drawing, height difference between versions can be found on page 1/2
- * presented spatial calibration will most likely differ from a true calibration of an assembled system, thus calibration should always be validated with e.g. *github.com/ethz-asl/kalibr.git*

Dept.	Technical reference	Created by	Approved by
		Christian Kaarre	
		Document type: Tech. Drawing	Document status: First Issue
		Title: CameraSystem IMUMount	DWG No.
			Rev. 00 / Date of issue Jun 2022 / Sheet 2/2