THOMAS

Eine universell einsetzbare Robotikplattform

Inhaltsverzeichnis

1 Informationen zum Projekt

1.1 Projektbeschreibung

Der Roboter in Größe eines Fahrradanhängers, der den Namen „THOMAS“ trägt, wird als Schülerprojekt am Kolleg St. Thomas Vechta schon seit mehreren Generationen entwickelt. Dieser besitzt unter anderem die Fähigkeit, sich mit Hilfe einer eigens entwickelten Steuerungssoftware manuell oder automatisiert fortzubewegen und dabei erkannte Wände und Hindernisse in einer Karte einzuzeichnen. Beitragend dazu sind die zwar etwas langsamen, aber dennoch sehr leistungsstarken Motoren, deren unabhängige Ansteuerung eine flexible Manövrierung gewährleisten, sowie ein handelsüblicher durchschnittlicher Desktopcomputer.

Durch diese Aspekte ergibt sich eine vielseitige Anzahl von Einsatzzwecken (Siehe Abschnitt 3), worauf dementsprechend unser Projektziel, „Eine universell einsetzbare Robotikplattform“ zu entwickeln, aufbaut.

1.2 Zielsetzung

Ein seitens der Schule gestelltes Jahresetat von 200€ fordert eine sorgfältige Auswahl sowie eine reichhaltige Recherche über entsprechende Komponenten und verhindert einen Spontankauf von Materialien. Des Weiteren wird reine Open-Source-Software, ob Betriebssysteme oder einfache Hilfsprogramme zur Analyse von Fehlern, genutzt, wodurch wir uns auch entschieden haben, unseren Programmcode unter eine Open-Source-Lizenz zu stellen und diesen auf der Entwicklungsplattform „Github“ zu publizieren, um eine reibungslose Zusammenarbeit zu ermöglichen. Dementsprechend bestehen wir auf einen gut lesbaren, durchdachten und strukturierten Programmiercode, was die Teamarbeit stark vereinfacht.

1.3 Team

1.3.1 Gründer

Wie anfangs erwähnt handelt es sich bei unserem Projekt um ein Generationenprojekt, an welchem schon im Jahr 2010 von Florian Thie, Lukas Bommes, Janek Ostendorf, Jan Wichelmann und später auch Nathan Wollek gearbeitet wurde. Diese haben sich um das Fundament des Roboters gekümmert, zu dem unter anderem die Umsetzung der Motorsteuerung und Aluminiumkonstruktion gehörte.

1.3.2 Nachfolger

Die Weiterentwicklung, Optimierung und bessere Vernetzung dieser Komponenten, sowie der Aufbau eines Sensorsystems und die Entwicklung mehrerer komplexer Steuerungs- und

Visualisierungsprogramme, wurde zur Aufgabe des nachfolgenden Teams, welches sich aus Marcus Wichelmann und Sören Busse zusammensetzt. Marten Herzog ist kürzlich dazugekommen und wird die Fortsetzung des Projektes im Rahmen der nächsten Generation sicherstellen.

2 Über „THOMAS“

2.1 Aufbau

Das Grundgerüst des Roboters besteht aus einer Aluminiumkonstruktion, deren Streben mit Hilfe eines Schiebemechanismus stabil verbunden sind. Drei der fünf Ultraschallsensoren befinden sich an der Front im 15-Grad-Winkel angeordnet, sodass ein möglichst breiter Bereich um den Roboter eingesehen werden kann. Die restlichen zwei befinden sich am Heck, wo auch ein entsprechender Hardwareschalter zur Unterbrechung des Motorenstroms für den Notfall angebracht ist.

Die beiden kettenbetriebenen Räder, die an ihrer Achse mit einem leichtem Handgriff zur Spannung der Kette bewegt werden können, werden durch ein drittes Stützrad unterstützt, was ein Umkippen des Roboters verhindert. An der linken Seite der Mainboardbefestigung ist gleichzeitig der Hauptschalter zur Unterbrechung der Stromversorgung montiert.

Auf einem Plexiglasgestell am hinteren Teil des Roboters finden sich zwei Infrarot Distanzsensoren, die, zusammen mit einer Kamera, auf einem Servo befestigt sind, ein Arduino mit einer aufgesteckten Steuerungsplatine sowie ein Infrarotempfänger, ein Display und eine Statusanzeige für die Qualität der WLAN-Verbindung.

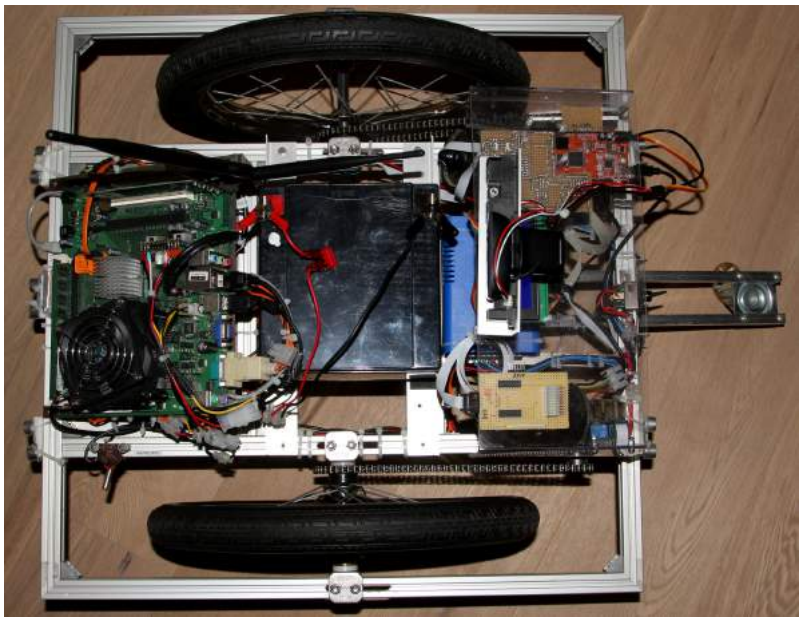

Abbildung 1: Draufsicht auf THOMAS

2.2 Hardwarekomponenten

2.2.1 Energiemanagement

Als Energiequelle wird ein GB 12-26 Bleiakku der regionalen Firma Panther mit einer Gesamtkapazität von 26Ah verwendet. Dieser beruht auf der AGM-Technologie, welche aufgrund des Nichteinsetzens flüssiger Säure das Gefahrenpotential minimiert und durch einen Tiefenentladeschutz das Eintreten einer dauerhaften Schädigung vermeidet. Zudem ist die Langlebigkeit des Akkus und der damit verbundene geringe Wartungsaufwand im schulischen Einsatz von Vorteil. Der durchschnittliche Ladezyklus des Akkus beträgt jedoch knapp 18 Stunden.

	Ausgeschaltet	Hochfahren	Leerlauf	Last
Spannung U in V	12,20	11,80	12,00	11,60
Strom I in A	0,00	2,40	1,10	2,60
Leistung P in W	0,00	28,32	13,20	30,16

Tabelle 1: Leistungsaufnahme des Computers

	Ausgeschaltet	Leerlauf	Last
Spannung U in V	0,00	12,16	12,08
Strom I in A	12,43	1,15	1,76
Leistung P in W	0,00	13,71	21,34

Tabelle 2: Leistungsaufnahme beider Motoren zusammen

2.2.2 Motorik und Motorsteuerung

Die ausgemusterten Scheibenwischermotoren eines alten Golfs geben dem Roboter die entsprechende Fortbewegungskraft und dienen einer flexiblen Manövrierung. Die Ansteuerung erfolgt durch eine Motorsteuerung „RN-MotorControl“ mit dem entsprechenden „RN-VNH2 Dualmotor“ Modul, die aus der 12V Gleichspannung zwei flexible, über RS232 manipulierbare PWM Signale erzeugt.

Eine Kraftübertragung auf die Räder erfolgt zunächst durch eine Schnecke, welche zwar die Geschwindigkeit verringert, aber die Kraft des Antriebes stark ausbaut, und anschließend über eine Kette, wie sie auch bei Fahrrädern aufzufinden ist.

2.2.3 Computerhardware

Anfangs erfolgte der Betrieb des Computers mit einem relativ schwachen Prozessor, der aber mit dem Anfang der Kameraübertragung schon bald ersetzt werden musste. Aktuell wird ein Intel Pentium Dual-Core E5400 eingesetzt, der alle rechenintensiven Vorgänge bewältigt und durch seinen geringen Energieverbrauch von 65 Watt punktet. Eine SSD

sichert dabei den reibungslosen Betrieb im Falle einer Erschütterung, bei der eine handelsübliche HDD die Stabilität des Systems stören würde.

Eine WLAN-Verbindung mit hoher Reichweite wird mit einer PCI-Karte mit zusätzlicher Antenne erreicht. Des Weiteren wird als Netzteil eine PicoPSU verwendet, welche immer eine konstante Ausgangsspannung für das Mainboard unabhängig von der Spannung des Akkus liefert und so die Kapazität des Akkus in Bezug zur Laufzeit optimal ausnutzt.

2.2.4 Sensorsystem

Das Sensorsystem basiert auf einem „Arduino 2560“, der über den USB-Anschluss auf Byte-Ebene mittels eines selbst erstellten Protokolls mit dem Computer kommuniziert und die entsprechenden Werte, nach Anfrage der Serversoftware, übermittelt. Als Sensoren werden fünf, aufgrund der Langen Messzeit per Parameter abschaltbare, Ultraschallsensoren sowie zwei in der Antwortzeit sehr schnelle Infrarot Distanzsensoren, die zur Kartierung dienen, verwendet. Dabei können diese Distanzsensoren mit Hilfe eines Servos um 180 Grad gedreht werden. Auf dem vierzeiligen LCD Display, dessen Menüführung per Fernbedienung bedient werden kann, werden Netzwerkstatus, Ultraschall-Messwerte und eventuell auftretende Fehlermeldungen angezeigt. Die Qualität der WLAN-Verbindung wird zusätzlich durch eine LED-Leiste in Echtzeit aktualisiert und visualisiert.

Hinsichtlich der hohen Kabelersparnis und Vielfalt der verfügbaren kompatiblen Module, haben wir uns außerdem dazu entschieden ein I2C-basiertes Bus-System zu verwenden, welches derzeit zur Anbindung von LCD-Display und Signalanzeige dient. Die direkte Ansteuerung der Status-LEDs wird durch zwei Module der gängigen PCF8574A-Baureihe, denen jeweils eine eigene Adresse zugewiesen wurde, ermöglicht.

2.3 Betriebssystem

Wie auch beim Code wird bei der Wahl des Betriebssystems auf Open-Source gesetzt. Wir haben uns für die Linux Distribution Debian in der Version 8/Jessie entschieden, da diese stabil und anpassbar ist und so eingerichtet werden kann, dass nur ein Minimum an nötiger Software zum Einsatz kommt und die Laufzeit des Akkus auf diese Weise gesteigert wird. Ein Upgrade auf die neueste Debian Version trug außerdem zu einem beschleunigten Startvorgang und einer besseren Kompatibilität mit der verbundenen Hardware bei.

2.4 Steuerungssoftware

2.4.1 Wahl der Programmiersprache

Die erste Version der Serversoftware basierte auf C++. Nachdem dieses allerdings immer instabiler wurde und strukturelle Probleme aufwies, die am Anfang des Projektes nicht

bedacht wurden, wurde die komplette Steuerungssoftware in der Sprache „Vala“ neugeschrieben. Diese ist besonders im Bereich der Desktopentwicklung für Gnome und Ubuntu verbreitet und besitzt eine Java ähnliche Syntax, die beim Vorkompilieren in C umgewandelt wird und deren Geschwindigkeit dementsprechend auch mit der von C vergleichbar ist. Dabei ist die Lernschwelle für Vala relativ gering, sodass sich auch die nachfolgenden Generationen schnell im Code zurecht finden und diesen einfach weiterentwickeln können. Die Clientsoftware dagegen wurde von vornherein auf Vala aufgebaut, da diese für die GUI-Programmierung in GTK-3 besondere Vorteile bietet.

2.4.2 Anforderungen

Besonders wichtig ist uns die Stabilität und Zuverlässigkeit der Client- und Serversoftware, deshalb haben wir uns unter anderem entschieden, diese neuzuschreiben. Dabei war ein fundamentaler Aspekt, dass die neue Serversoftware mit entsprechenden Parametern ausgestattet wird, um beispielsweise gesonderte Module, welche bestimmte Hardwarebausteine voraussetzen, zu deaktivieren, da so eine Lauffähigkeit und Testbarkeit außerhalb der Robotikplatform ermöglicht wird. Sie sollte zudem modular aufgebaut sein, sodass eine Implementierung zusätzlicher Funktionen vereinfacht wird.

2.4.3 Funktionsweise

Der Roboter basiert auf dem simplen Client-Server-Prinzip, wobei hier der Roboter als Server fungiert. Die ursprüngliche Kommunikation basierte auf einem TCP Socket, der aber nach dem Neuschreiben des Codes durch den Freedesktop Standard DBus ersetzt wurde (Siehe Abschnitt 2.4.4). Die Informationsübermittlung zur Motorsteuerung erfolgt über einen RS232 Port. Dazu werden Datenpakete, die jeweils mit dem doppelten Char 35 (ASCII: #) beginnen, gefolgt von einem Kommandobyte (Siehe Abschnitt 6.1) und weiteren Parametern, gesendet, wodurch entsprechende Aktionen, wie dem Steuern der Motoren oder dem Auslesen der aktuellen Stromstärke, durchgeführt werden.

Die Implementierung dieses Protokolls ermöglicht zugleich eine beschleunigte Bewegung, aber auch das Setzen der Geschwindigkeit direkt, um beispielsweise einen Notstopp auszuführen. Auch die Relaiskarte, die eine von der Clientsoftware ausgelöste Hupe ansteuert, kommuniziert über RS232 und nimmt entsprechende Schaltbefehle der Serversoftware entgegen, wobei hier ein USB-Adapter mangels zweitem RS232 Ports genutzt werden musste.

Das Kamerabild wird mittels OpenCV in der Serversoftware aufgenommen, woraufhin es je nach Qualitätseinstellung konvertiert und als Bytestream über UDP an alle verbundenen Clients übertragen wird. Durch diese Methode wird eine Latenz von unter 100ms sowie eine individuelle Anpassung an die zur Verfügung stehenden Netzwerkressourcen durch die im laufenden Stream einstellbare Übertragungsqualität gewährleistet. Zugleich werden im Client alle wichtigen Systeminformationen wie CPU, RAM oder Netzwerkauslastung

angezeigt, welche dank entsprechender Bibliotheken einfach am Server ausgelesen werden können.

2.4.4 Schnittstellen zur Kommunikation

Die wichtigste Schnittstelle zur Kommunikation ist die des Clients mit dem Server. Diese basiert auf dem Freedesktop Standard „DBus“, welches aufgrund der vielzahligen Möglichkeiten besonders in der Desktop-Entwicklung für Linux verbreitet ist. Mit leichten Anpassungen kann dieses Protokoll allerdings direkt über das Netzwerk verwendet werden, wodurch am Client/Server ein direkter Aufruf einer Methode ohne eines eigens erstellten, separaten TCP Sockets möglich ist. Für die meisten Programmiersprachen ist deshalb auch eine Bibliothek vorhanden, sodass eine Entwicklung der Clientsoftware auf unterschiedlichen Plattformen möglich wäre.

Eine weitaus plattformunabhängigere Lösung haben aber wir mit der parallel verwendbaren Implementierung von Websockets erreicht. Dazu werden JSON-Anfragen an den Server gesendet, der die angefragten Funktionen dann ausführt und entsprechend antwortet. Dadurch wäre auch eine einfache Implementierung einer im Webbrowser laufenden Software möglich, die somit eine deutliche Erweiterung der möglichen Plattformen bietet. Ferner ist die Kommunikation mit dem Arduino und der damit verbunden Sensorik über eine Infrarotfernbedienung möglich, die zur Navigation des Menüs auf dem Display oder dem Notaus verwendet wird.

2.5 Kartierung

Dank der frei drehbaren Entfernungssensoren ist es dem Roboter möglich Messreihen aufzustellen, die Aussagen über die Umgebung vor dem Roboter zulassen. Die Auswertung dieser, wird von einem eigens für diesen Roboter entwickelten Algorithmus übernommen, der in den Messdaten nach Regelmäßigkeiten, die auf Wände oder Hindernisse hinweisen, sucht und die Motorik passend ansteuert. Dadurch, dass der Roboter dem Backtracking-Prinzip folgt, kann jeder Bereich des Raumes erfasst und bestmöglich in einer Karte eingezeichnet werden. Eine Herausforderung für die Entwicklung dieses Kartierungsalgorithmus bestand vor allem darin, die meist relativ ungenauen Sensorwerte zu filtern, zielführend zu interpretieren und dafür zu Nutzen mögliche Fehler, die durch die nicht durch Odometrie-Sensoren überprüften Motoren auftreten, zu korrigieren.

3 Denkbare Einsatzzwecke

3.1 Gartenhelfer

Es gibt viele Leute, die aufgrund ihres fortgeschrittenen Alters nicht mehr die Kraft besitzen, die Gartengeräte, wie zum Beispiel Eimer oder Schubkarre, zu tragen. Dennoch möchten diese gerne in der Natur tätig sein und ihren Garten pflegen. Genau dort könnte ein Einsatzbereich unserer Roboterplattform sein. Aufgrund der stabilen Bauweise und der leistungsstarken Motoren, könnte auf diesem ein Schubkarrenbehälter befestigt werden, sodass THOMAS einfach per Joystick an den gewünschten Ort navigiert werden oder wahlweise über die zukünftig geplante automatische Routenfunktion eine vordefinierte Strecke mit Hilfe der Karte abfahren kann. In Zukunft soll dieser zudem über Infrarot Tracking-Punkte, die z.B. an Hosen oder Rollatoren befestigen werden könnten, einer Person folgen, was besonders bei der Gartenarbeit von Interesse wäre.

3.2 Kehrroboter

Auf Anreiz unseres mysophobischen Hausmeisters könnte man THOMAS auch als Kehrroboter einsetzen. Dazu könnte eine Kehrmaschine, die aufgrund der starken Motoren mit Leichtigkeit gezogen werden würde, hinter den Roboter gehangen werden und somit eine automatische Reinigung nach vordefinierten Mustern ermöglichen.

3.3 Kartierungsroboter

Möchte man beispielsweise eine Karte von einem Gebäude erstellen, so muss dies oft in mühsamer Handarbeit gezeichnet werden. Diesen Job könnte THOMAS erledigen, da er durch seinen Backtracking-Algorithmus automatisch den Weg durch das Gebäude findet und alle Wege und Wände dementsprechend dokumentiert. Des Weiteren könnten in diese Karte sowohl die WLAN-Signalstärken als auch die Mobilfunkanbindung innerhalb eines Gebäudes eingezeichnet werden.

4 Mögliche Anknüpfungspunkte

Odometrie Zur weiteren Verbesserung des Roboters und Erweiterung der möglichen Einsatzzwecke, wurden Überlegungen über weitere Anknüpfungspunkte angestellt. Diese umfassen beispielsweise eine Erhöhung der Präzision der Motorik durch optische Odometrie-Sensoren, welche an der Motorsteuerung angebunden werden können.

Computerhardware Eine Optimierung der Stromaufnahme und Rechenleistung des Roboters könnte zudem durch eine neuere CPU oder einen, anstelle des Mainboards verwendeten, Einplatinencomputer mit ARM-Architektur ermöglicht werden. Im Zuge der Umsetzung einer der oben beschriebenen Ideen (Siehe Abschnitt 3), sollte zudem auch über den Bau einer Ladeschale entschieden werden, welche dem Roboter das selbstständige Wiederaufladen bei kritischem Akkuzustand erlauben würde.

Touchscreen Durch die Nachrüstung eines kleinen Touchscreens könnte des Weiteren die Benutzerschnittstelle um Längen verbessert und intuitiver gestaltet werden. Eine präzisere Fehleranalyse, eine Anzeige des Kamerabildes oder eine manuelle Steuerung der Motoren und Überprüfung der Sensorwerte wäre mit dieser Komponente leicht umsetzbar. Zudem würde die Einrichtung neuer WLAN-Netzwerke nicht länger den Anschluss eines LAN-Kabels oder eines Bildschirms samt Tastatur benötigen.

5 Weiterführende Links

THOMAS-Projekt Webauftritt: `http://thomas-projekt.de`

Github: `https://github.com/THOMAS-Projekt/`

YouTube Kanal: `https://www.youtube.com/user/THOMASProjekt`

Dokumentation des Motorcontrollers: `http://www.robotikhardware.de/download/rnmotorcontrol.pdf`

6 Anhang

6.1 Datenblatt

OS	
Typ	x86_64 GNU/Linux
Distribution	Debian
Version	8
Codename	Jessie
Kernel	3.16.0-4-amd64
eigene Software	
THOMAS-Viewer (Kamerabild und Steuerung)	
THOMAS-Server (C++ und Vala)	
THOMAS-Client (C++ Steuerung)	
THOMAS-Sensorik (Arduino)	
THOMAS-Simulator	
Motherboard	
Typbezeichnung	Fujitsu D2950-A11 GS 2
Sockel	Sockel 755
CPU	
Typbezeichnung	Pentium(R) Dual-Core CPU E5400
Kerne	2
Kerntakt	2,7 Ghz
TDP	65 W
Netzteil	
Typbezeichnung	PicoPSU M3-ATX
Eingangspannung	6 V - 24 V DC
Ausgangsleistung	125 W
Ram	
Typbezeichnung	M378T5663QZ3-CF7
Typ	DDR2-RAM
Größe	2GB
Frequenz	800 MHz
SSD	
Typbezeichnung	SAMSUNG SSD 830
Typ	Solid State Drive (SSD)
Größe	64 GB
Formfaktor	6,4cm (2,5")"
Transferraten (Sata 2)	252 Mbps (Lesen), 249 Mbps (Schreiben)

Transferraten (Sata 3)	520 Mpbs (lesen), 320 MBps (Schreiben)
WLAN Karte	
Typbezeichnung	TP-LINK TL-WN851ND
Übertragungsstandard	11n 300Mbit/s
Webcam	
Typbezeichnung	Logitech C270
Auflösung	720p
Akkumulator	
Typbezeichnung	Panther GB 12-26
Technologie	AGM
Nennspannung	12V (real 13,8V - 11V)
Kapazität	26Ah
Masse	8kg
Motorsteuerung	
Typ	RN-MotorControl + RN-VNH2 Dualmotor
Funktionsweise	PWM-Steuerung mit MOSFET H-Brücke
Schaltspannung	6V - 16V
Schnittstelle	RS-232 TTL Pegel, I2C, RC Empfänger
Relaiskarte	
Typbezeichnung	Conrad USB Relaiskarte
Kanäle	8
Schaltleistung	172W (24V DC, 7A)
Stromaufnahme	300mA
Schnittstelle	conrad Adapterplatine USB 2.0 zu RS232
Arduino	
Typbezeichnung	Arduino Mega 2560
Speicher	256KB
Schnittstelle	USB 2.0
I/O Pins	54 Digital, 16 Analog

6.2 Bilder

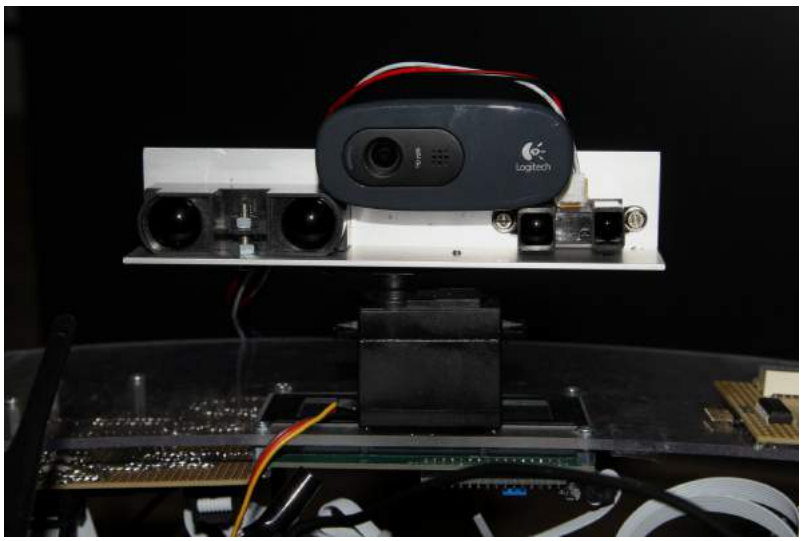

Abbildung 2: Drehbare Abstandssensoren und Kamera

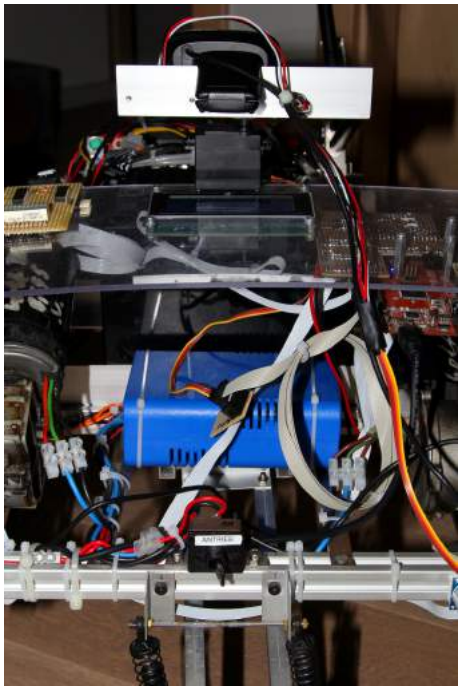

Abbildung 3: Ansicht von hinten

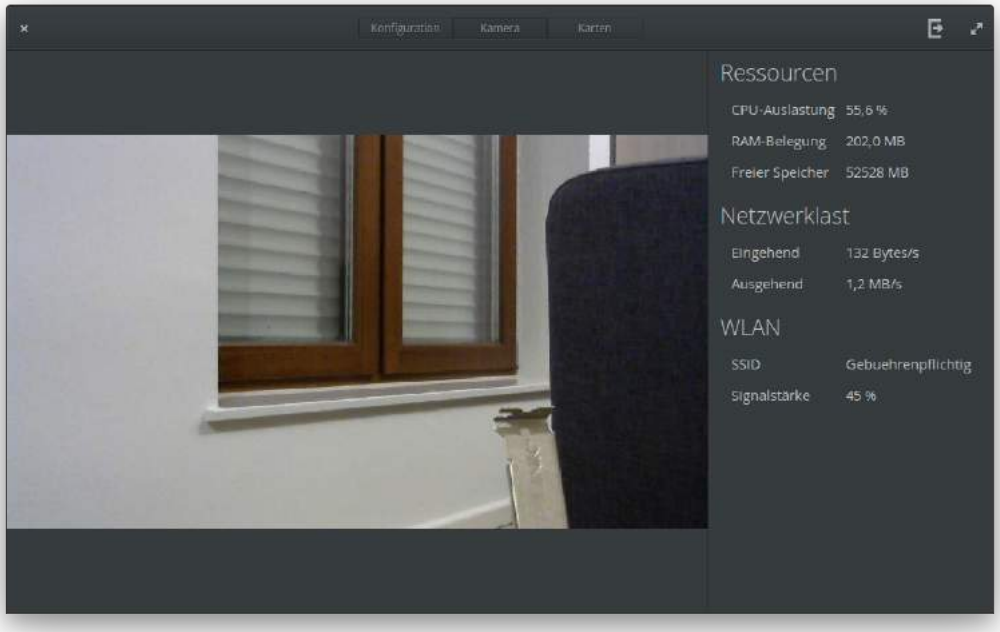

Abbildung 4: Anzeige des Kamerastreams samt Telemetrie im Steuerungsprogramm

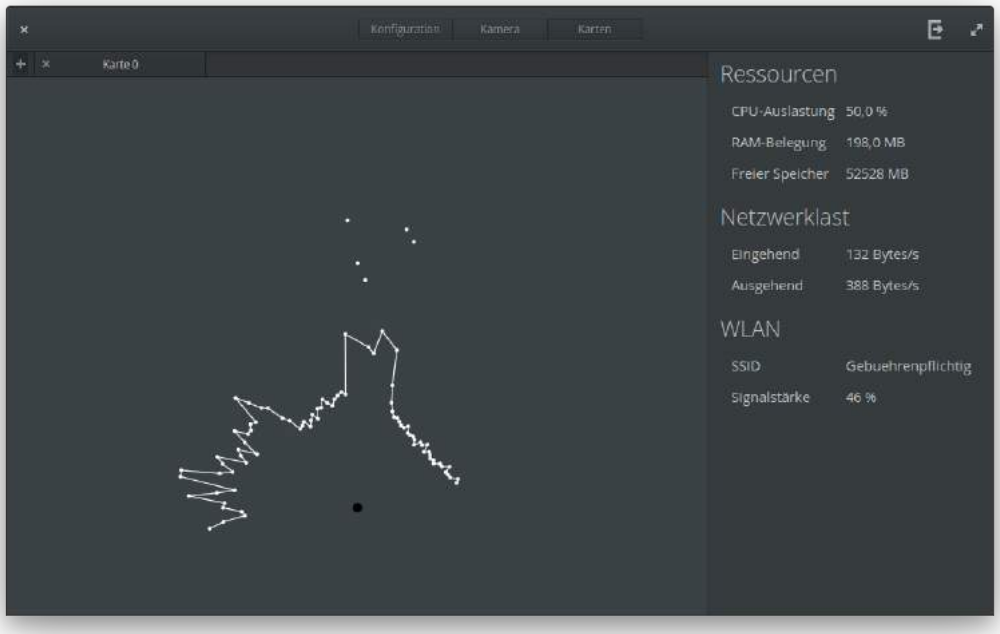

Abbildung 5: Visualisierung einer erfassten Messreihe