

Conference Paper Title*

*Note: Sub-titles are not captured in Xplore and should not be used

1st Katrin Glöwing
Interaktionstechnik und Design
Hochschule Hamm-Lippstadt
Lippstadt, Deutschland
Immatrikulationsnummer: 2170348

2nd Domenic Drechsel
Interaktionstechnik und Design
Hochschule Hamm-Lippstadt
Lippstadt, Deutschland
Immatrikulationsnummer: 2170168

3rd Justin Frommberger
Interaktionstechnik und Design
Hochschule Hamm-Lippstadt
Lippstadt, Deutschland
Immatrikulationsnummer: 2170394

4th Alexander Wilms
Interaktionstechnik und Design
Hochschule Hamm-Lippstadt
Lippstadt, Deutschland
Immatrikulationsnummer: 2150884

Abstract—Overview of document including main outcomes

I. MOTIVATION

In vielen Fällen ist es für Menschen lebensbedrohlich Personen zu bergen.

Bei Naturkatastrophen wie Waldbränden, Tsunamis beziehungsweise generell Überflutungen oder Vulkanausbrüchen ist es zu gefährlich menschliche Bergungsteams einzusetzen. In diesen Fällen ist der Rescue Robot hilfreich.

Rescue Robots die in brennenden Waldgebieten oder nach Vulkanausbrüchen eingesetzt werden sind hitzebeständig. Robots die bei Tsunamis und Überflutungen zum Einsatz kommen sind wasserbeständig und können sowohl auf Wasser fahren, wie auch auf Land. Diese sind auch bei Schiffsunglücken hilfreich.

Zudem können Rescue Robots auch für Reparaturen an Atomkraftwerken und bei Unfällen an Atomkraftwerken verwendet werden, da dies aufgrund der hohen Strahlung auch nur begrenzt für Menschen möglich ist.

In diesem Projekt wird das im nächsten Absatz folgende Szenario betrachtet.

[Katrin Glöwing]

A. Szenario

Der Rescue Robot findet in diesem Projekt Verwendung nach einer Explosion im Mehrfamilienhaus zur Bergung einer verletzten Person.

Figur 1 zeigt die Testumgebung. Das Mehrfamilienhaus ist gekennzeichnet durch die "Non-traceable ground" Fläche in der Mitte der Karte. Der Robot startet von unten links auf der Karte und die Person (als roter Blitz gekennzeichnet), welche geborgen werden soll, befindet sich unten rechts auf der Karte. Zur Orientierung gibt es vier Signalposten (gekennzeichnet als "audio and/or light signal"), die den Robot leiten.

Identify applicable funding agency here. If none, delete this.

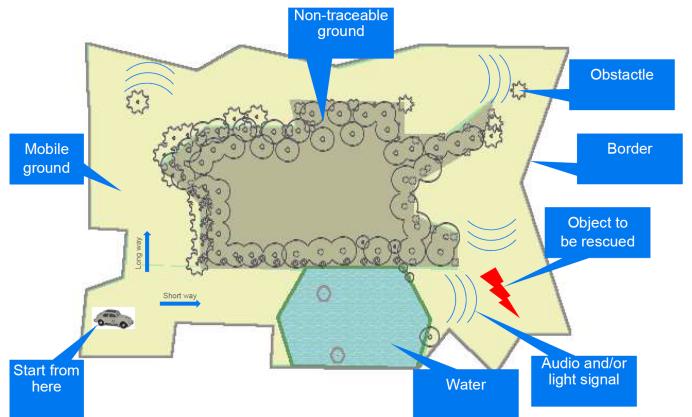


Fig. 1. Testumgebung

Der Untergrund kann Stein (beiger Bereich) und Wasser (blauer Bereich) sein. Die Kreise im und am Wasser sind teilweise noch brennende Gegenstände, die der Robot aus dem Weg räumen muss.

Der Rescue Robot hat die Möglichkeit einen von zwei Wegen zu wählen ("long way", "short way"). Die Richtungen sind durch die Pfeile unten rechts gekennzeichnet. Der kurze Weg führt den Robot durch das Wasser zu der zu bergenden Person und wieder zurück durch das Wasser. Der lange Weg führt den Robot einmal um das Mehrfamilienhaus herum zu der Person und wieder zurück um das Haus zum Startpunkt.

[Katrin Glöwing]

B. Anforderungen

Figur 2 zeigt die Anforderungen des Projektes. Funktionale Anforderungen (Typ: F) sind Anforderungen die beschreiben was das System oder das Produkt ausführen muss [1] und nicht-funktionale Anforderungen (Typ: NF) beschreiben die fundamentale Basis der Systemarchitektur [2].

Type	ID.	Description
F	010	Der Roboter muss Hindernisse erkennen.
F	011	Der Roboter muss den Abstand zu den Hindernissen erkennen.
NF	012	Der Roboter muss mit acht Sensoren ausgestattet sein.
NF	013	Die Sensoren müssen in gleiche Abständen am Roboter angebracht sein (N, NO, O, OS, S, SW, W, WN).
F	014	Der Roboter muss Hindernisse entfernen.
F	020	Der Roboter auf Land fahren können.
F	030	Der Roboter muss durch Wasser fahren können.
NF	031	Der Roboter muss einen Propeller besitzen.
F	041	Der Roboter muss erkennen ob er auf Wasser oder Land fährt.
NF	042	Der Roboter muss einen Feuchtigkeitssensor besitzen.
F	043	Der Roboter muss für das Fahren durch das Wasser den Propeller verwenden.
F	044	Der Roboter muss für das Fahren auf dem Land die Ketten verwenden.
NF	050	Der Roboter muss einen Kompass besitzen.
F	060	Der Roboter muss einen Akku besitzen.
F	070	Der Roboter muss mit dem "Umweltsystem" interagieren.
F	080	Der Roboter muss gut sichtbar sein.
NF	090	Der Roboter muss mit vier Lichtquellen ausgestattet sein.
F	110	Der Roboter muss Personen identifizieren können.
NF	111	Der Roboter muss eine Kamera haben.
F	112	Der Roboter muss nach Erkennung einer Person die Livesteuering aktivieren.
NF	113	Der Roboter muss über einen Roboterarm verfügen (für die Livesteuering).
F	120	Der Roboter muss die Richtung in die er fahren möchte anzeigen.
NF	121	Der Roboter muss mit einem Display ausgestattet sein.
NF	122	Das Display muss die nächste Richtung (mit Pfeilen) anzeigen.

Fig. 2. Anforderungen

Das Projekt ist die Simulation eines Rescue Robots.

Zusammengefasst sind die Anforderungen folgende:

- Der Roboter muss Hindernisse erkennen und ausweichen.
- Der Roboter muss auf Land und Wasser fahren können.
- Der Roboter muss seinen Standort weitergeben.
- Der Roboter muss anzeigen in welche Richtung er als nächstes fährt.
- Der Roboter muss Personen identifizieren.
- Der Roboter muss einen Roboterarm besitzen, der per Liveschaltung verwendet wird um Lebewesen und Objekte zu bergen.

[Katrin Glöwing]

II. SKETCH OF APPROACH

A. Kontextabgrenzung

Figur 3 zeigt das Kontextdiagramm. In der Mitte des Diagramms steht das System (blau) selbst.

Die ersten Überlegungen haben ergeben, dass die physikalische Umgebung eine Rolle spielen wird. Der Untergrund könnte Wasser, Erde, Stein, Sand, etc. sein und dies kann mithilfe von Sensoren ermittelt werden.

Außerdem sind das Wetter und die Temperatur eventuell auch zwei wichtige Komponenten. Inwiefern muss der Roboter wetter beständig sein und auch hitze- und/oder kältebeständig. Des Weiteren sind auf der Karte noch die Signalposten aufzufinden, die Audio-/Lichtsignale senden. Hier besteht auch eine Verbindung zum Rescue Robot, da die Signale ihn über die Karte führen.

Das Bergungsobjekt gehört auch in den Kontext. Der Roboter soll das Bergungsobjekt mithilfe von Sensoren und einer Kamera erkennen.

Zudem gibt es noch einen internen und einen externen Controller. Der externe Controller ist mit dem Rescue Robot durch das Human Machine Interface (HMI) verbunden. Sobald der Roboter am Bergungsobjekt angekommen ist, soll sich das HMI anschalten und der Roboterarm soll durch einen externen Controller, per Livesteuering, gesteuert werden.

Die Hindernisse teilen sich auf in Wände und Objekte. Den Wänden soll der Roboter ausweichen und die Objekte, welche in Abschnitt I-A beschrieben sind, sollen bis zu einer bestimmten Größe überfahren und ab einer bestimmten Größe umfahren werden.

So waren die ersten Überlegungen der Gruppe zum Kontext.

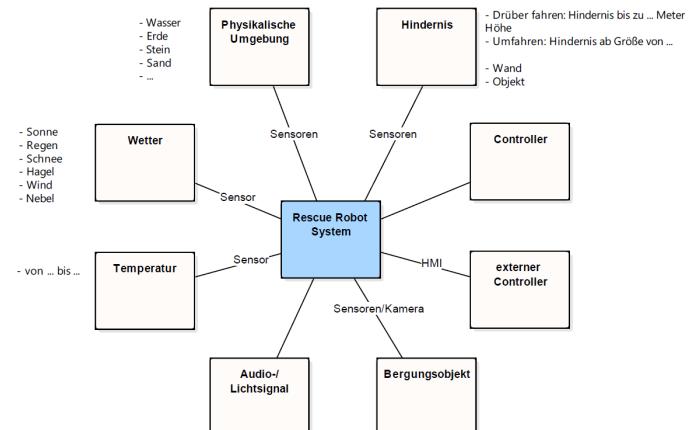


Fig. 3. Kontextdiagramm

[Katrin Glöwing]

B. Szenarienvielfalt und Anforderungsunterschiede

In dem Modul 'Projekt angewandte Elektrotechnik' bestand die Aufgabe darin ein Rescue Robot zu konzipieren. Dabei handelt es sich um einen Roboter, der in Gefahrensituationen interagieren kann. Die Vorteile von einem autonomen oder auch ferngesteuerten Roboter sind groß. So kann dieser sich in Situationen begeben, die für einen Menschen oder einen Rettungshelfer gefährlich sind. Dabei sind die Aufgabenanforderungen und die mögliche Szenarien-Vielfalt quasi unendlich.

[Alexander Wilms]

C. Szenarienvielfalt

Mögliche Szenarien sind zum Beispiel Umweltkatastrophen. Nach einem Tsunami oder schweren Regenfällen sind Überschwemmungen die Folge. Dabei können Erdmassen mitgetragen werden und neu entstandenen Flüsse in urbanen Gebieten entstehen. Wenn die Luftrettung oder eine generelle Rettungsmaßnahme von Rettungshelfern scheitert durch zu gefährliche Bedingungen, könnte ein Roboter helfen. Dieser kann an Orte gelangen die unzugänglich sind und den Kontakt

zu Personen aufstellen bzw. einen Rettungsversuch einleiten. Dabei wäre er so konzipiert das dieser schwimmen kann und sich durch unklares fließendes Gewässer navigiert. Als contraire Szenario ist aber auch ein Waldbrand möglich oder auch ein Industriebrand. Durch die vorherrschende Hitze und hohe Temperatur sind ganz andere Anforderungen an den Rescue Roboter gestellt. So müsste dieser an Land und über Hindernisse fahren können und hitzebeständig sein. Es sind aber auch noch andere Szenarien möglich, die für Rettungskräfte zu gefährlich sind. Dazu zählen zum Beispiel Chemie oder Atomunfälle. Wenn die Strahlung zu hoch ist, hat dies langanhaltende Folgen für Menschen. Ein Roboter könnte in die Gefahrenzone hineinfahren. Über Kameras und Messinstrumente können wichtige Daten weitergeleitet werden. Hier müsste der Roboter speziell beschichtet sein, um der Strahlung standzuhalten.

[Alexander Wilms]

D. Anforderungsunterschiede

Es zeigt sich, dass es viele mögliche Szenarien für einen Rescue Roboter gibt. Dabei sind die Anforderungen höchst unterschiedlich und situationsbedingt. Jedes Einsatzgebiet erfordert andere Maßnahmen und technische Umsetzungen. Wünschenswert wäre ein Roboter, der alles kann. Da dies aktuell und zukünftig technisch nicht realisierbar ist, ist es sinnvoll sich auf ein Szenario zu fokussieren. Durch die Spezialisierung wird der Roboter bestmöglich ausgestattet, um die Aufgaben zu erfüllen.

[Alexander Wilms]

III. BLOCKDIAGRAMM

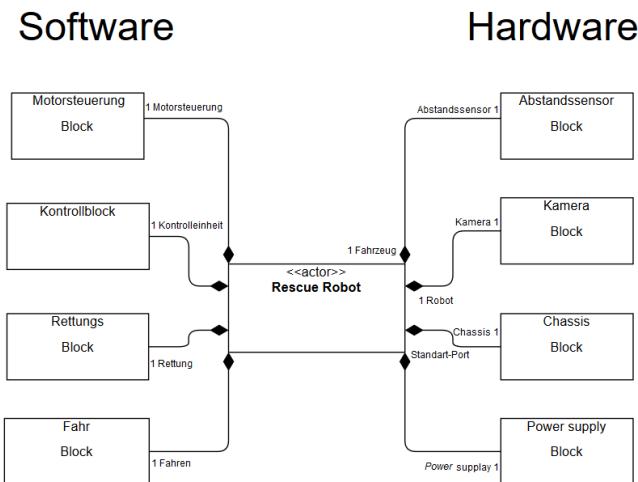


Fig. 4. Blockdiagramm

Auf Fig. 3. wird ein Blockdiagramm dargestellt. Mit einem Blockdiagramm werden die einzelnen Komponenten des Systems beschrieben. Der Hauptblock ist der Rescue Robot Block. Dieser Block ist verbunden mit den einzelnen Unterblöcken.

Die Blöcke sind aufgeteilt in Software Blöcke und Hardware Blöcke. Zudem sind die Blöcke mit dem Rescue Robot Block verbunden, da diese zusammenhängend sind.

[Justin Frommberger]

IV. STAKEHOLDER

Für die Planung des Projektes, analysierten wir vorab die möglichen Stakeholder, die im Laufe des Projektes mit dem Roboter in Kontakt kommen. Stakeholder sind Menschen, die in irgendeiner Form Anspruch auf ein Projekt haben und das kann aus unterschiedlichen Gründen begründet sein. Weil sie Teil des Projektteams sind oder weil sie sich für das Projekt interessieren. Im Anschluss listeten wir dann alle möglichen Stakeholder auf und analysierten ihre möglichen Anliegen an dem Projekt.

[Justin Frommberger]

A. Politiker

Der Rescue Robot hat Kontakt mit der Umwelt und Personen. Dies kann zu rechtlichen Problemen führen.

Durch Erweiterungen am Rettungswagen, könnte so die Umwelt verbessert werden und die möglichen Risiken verhindert werden.

[Justin Frommberger]

B. Menschen, die gerettet werden

Die Menschen sind nicht einverstanden mit der neuen Technik und möchten diese nicht in Anspruch nehmen. Durch Sicherheitsverifizierungen und Informationen über den Roboter fühlen sich die Menschen sicherer und unterstützen das Projekt.

[Justin Frommberger]

C. Bediener

Der Bediener des Roboters kennt sich mit der neuen Technik nicht aus, ist deshalb eher skeptisch. Durch Angebote von Weiterbildungen für den Roboter können die Bediener so begeistert werden mit der neuen Technik zu arbeiten.

[Justin Frommberger]

D. Monteure

Der Monteur kennt sich mit der neuen Technik nicht aus, ist deshalb eher skeptisch. Durch zeigen von Bauplänen und Hilfestellungen können die Monteure begeistert werden.

[Justin Frommberger]

V. USECASE

Aus unseren Anforderungen haben wir ein Use Case Diagramm erstellt. Wir recherchierten und stellten uns folgende Fragen: "Was wollen wir und was muss die Lösung des Projektes sein?"

Auf Fig. 1. sieht man das erstellte Use Case Diagramm. Zu sehen sind die drei Akteure die als Mensch dargestellt werden. Diese befinden sich außerhalb des Systems (Die Rettung eines

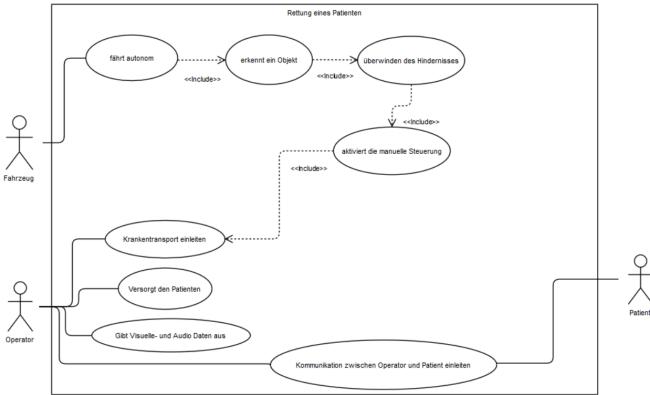


Fig. 5. UseCase

Patienten). Die Akteure sind mit dem Use Cases verbunden. Jeder Akteur hat unterschiedliche Aktivitäten wie, dass der Rescue Robot autonom fährt. Die einzelnen Use Cases sind untereinander verbunden über include Beziehungen.

Beispiel Ablauf: Der Rescue Robot fährt autonom. Dies included, dass ein Objekt erkannt wird. Anschließend wird das Hindernis überwunden, indem zu der manuellen Steuerung gewechselt wird. Daraufhin leitet der Akteur "Operator" den Krankentransport ein, versorgt den Patienten und gibt ihm Visuelle- und Audiodaten aus. Zum Schluss folgt die Kommunikation zwischen dem Akteur "Operator" und dem Akteur "Patienten."

[Justin Frommberger]

A. Aktivitätsdiagramm

Für jedes Use Case wurde daraufhin ein Aktivitätsdiagramm erstellt. Beispiel ist Fig. 2. Hier zu sehen ist das Aktivitätsdiagramm des autonomen Fahren. Das Diagramm beschreibt die Möglichkeiten vom autonomen Fahren. Der Sensor wird ausgelesen, anschließend wird dann entschieden, ob der Rescu Robot links, rechts oder geradeaus lenkt. Zum Schluss wird bestimmt, ob der Robot vorwärts oder rückwärts fährt.

[Justin Frommberger]

VI. CONCEPT PART

A. Paperprototyp

Mit der Festlegung des Szenarios sind die ersten Ansätze entstanden. Für das erste Design wurde die Methode des Paper Prototyp genutzt. Dabei handelt es um eine handgezeichnete Skizze, die genutzt wird, um erste Ideen visuell darzustellen. Hierbei können einfache Prozesse, Umrisse und Komponenten frei eingezeichnet werden. Im Design Thinking wird diese Art des Low-Fidelity-Prototyping gerne eingesetzt da der Prozess schnell und einfach ist, um die Grundkonzepte festzuhalten.

Auf der ersten Paper Prototyp Skizze sind zwei Perspektiven des Roboters dargestellt. Die obere Perspektive zeigt die Ansicht von Oben auf den Roboter. Die untere zeigt die seitliche

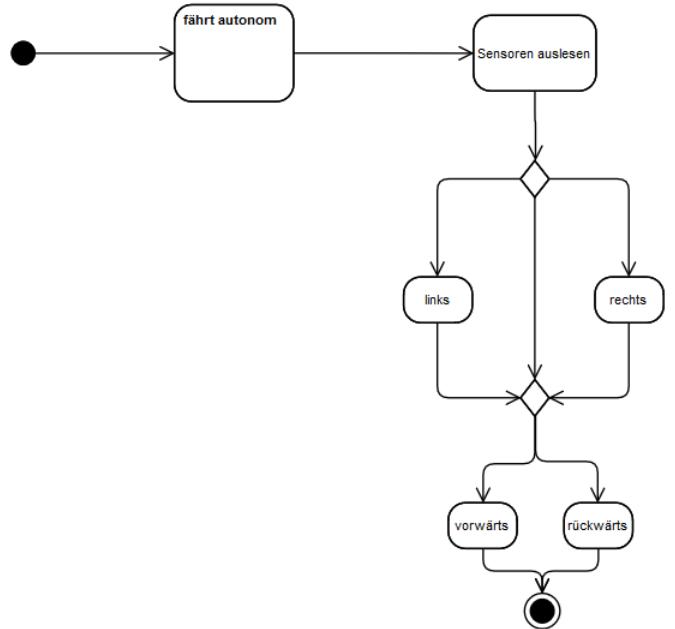


Fig. 6. Aktivitätsdiagramm

Perspektive. Auf den ersten Blick ist zu erkennen, dass es sich um ein Hybriden Roboter handelt. Dieser soll durch die Skizze schon zeigen, dass das Fahren auf dem Land und auch auf dem Wasser möglich ist. Auffällig ist die spitzzulaufende Form im vorderen Bereich. Die Bug-Form ähnelt der eines Boots oder Schiffs. Zudem erkennt man im hinteren Bereich, unten angebracht eine Schiffsschraube. Für den Landbetrieb sind Räder eingezeichnet. Die vier Räder befinden sich im unteren Bereich. Der Roboter soll die Möglichkeit haben Hindernisse wegzuräumen und Personen zu helfen. Dafür wurde ein Greifarm auf dem Roboter eingezeichnet. Dieser Greifarm kann vielseitig eingesetzt werden. Auf der Ansicht von Oben sind einige Details skizziert. So befinden sich dort insgesamt acht Abstandssensoren in verschiedenen Richtungen. Im hinteren Bereich ist die Motorsteuerung eingezeichnet, mittig ein Akku als Energieversorgung und vorne eine Logikkomponente. Im vordersten Bereich ist eine Wärmebildkamera eingeplant für den Einsatz in Hitzebereichen. Die Skizze zeigt somit die wichtigsten Komponenten und die Grundidee. Die Visualisierung stellt dar wie der Landbetrieb und Wasserbetrieb grob funktionieren soll. Zudem wird die Idee eines multifunktionalen Greifarms dargestellt.

[Alexander Wilms]

B. Erste 3D-Modellierung

Im Folgenden ein erstes Rendering des Prototyps in SolidWorks erstellt. Dabei handelt es um ein CAD Programm welches rechnerunterstützte Konstruieren ermöglicht. Das Modell wird 3-dimensional dargestellt. So ist ein räumliches Verständnis möglich und die grobe Form wird erkennbar. Auf dem Rendering zeigt sich die Bug-Form vorne, unten die vier

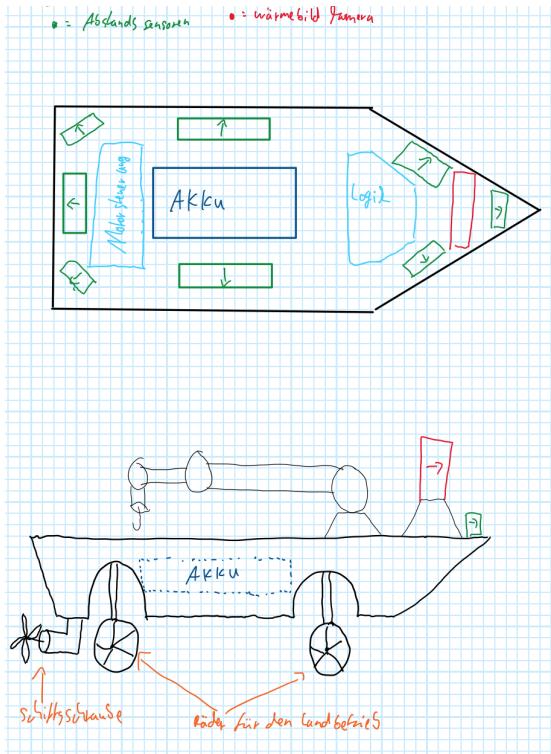


Fig. 7. Erster Paper Prototyp

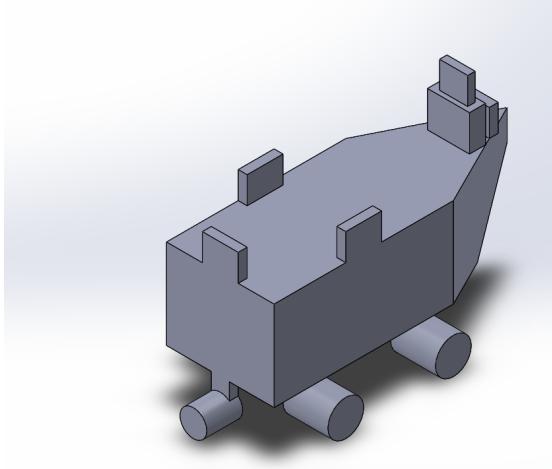


Fig. 8. Erstes CAD-Modell

Räder und hinten die Schiffsschraube. Im oberen Bereich sind erste Sensoren angebracht, sowie im vorderen Bereich eine Kamera. Der Greifarm wie er auf der Skizze zu sehen ist, ist nicht im Rendering. Das 3D-Rendering stellt die grobe Bauweise und Form vereinfacht dar.

[Alexander Wilms]

C. Reale Beispiele

Im Verlauf des Projekts wurde das Szenario genauer untersucht und analysiert. Die Use Case Anforderungen wurden



Fig. 9. Ladekran



Fig. 10. Löschpanzer

definiert. Das Design ist mit den Anforderungen gewachsen und hat sich teilweise verändert. Da es sich um ein echtes Szenario halten könnte, wurde der Blick auf Beispiele aus dem echten Leben gerichtet. Die Recherche zeigte das unsere Ideen schon teils real umgesetzt wurden. So gibt es LKWs mit Ladekränen die auch schwere Lasten heben können. Dabei ist der Vorteil, dass der Kran bzw. Greifarm durch den LKW flexibel überall hingefahren werden kann, je nach Einsatzgebiet. Die Hydraulik ermöglicht dem Greifarm einen großen Radius abzudecken und verschiedene Höhen und Winkel anzusteuern. Des Weiteren ergab die Recherche das die Feuerwehr teils Löschpanzer nutzt, welche auf Ketten fahren und aus einem feuerfesten Material sind. Die Recherche ergab das es bestimmte Kettenantriebe gibt, die auch für höhere Steigungen geeignet sind. Zudem eignen sich Ketten wie an dem Löschpanzer auch für höhere Temperaturen da diese nicht schmelzen. Der Ausblick zeigte das bereits echte Fahrzeuge existieren und die Komponenten auf einen neuen Rescue Roboter angewendet werden können.

[Alexander Wilms]

D. Prototyp mit Kettenantrieb

Im weiteren Verlauf erfolgte eine weitere Paper Prototyp Skizze. In der Recherche ergab sich, dass es für bestimmte Hitze und Feuersituationen spezielle Löschpanzer gibt. Die Panzerkettenansatz wurde übernommen. Bei einer erhöhten Temperatur besteht die Gefahr, dass gummierte Reifen schmelzen können und so ein Rettungsvorgang nicht mehr möglich ist. Um sicher zu sein, wurde das Design des Roboters etwas verändert und statt Rädern folgte ein Kettenantrieb. Dieser hat zu dem Zeitpunkt einen Kettenantrieb mit kleinen Schaufeln, um sich im Wasser fortzubewegen als temporäre Alternatividee. Die Schiffsschraube ist nicht auf dieser Skizze aber ist im weiteren Verlauf mit eingeplant für einen noch besseren Wasserforttrieb. Auf dieser Skizze



Fig. 11. Kettenfahrzeug

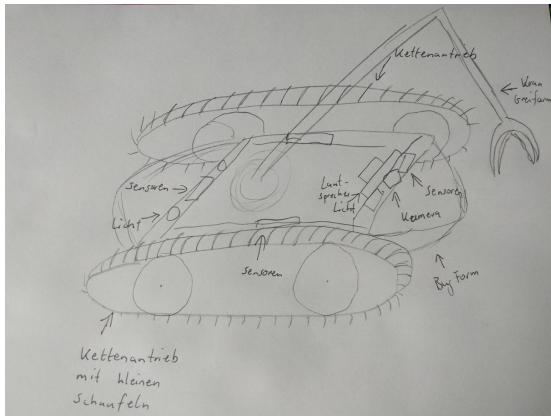


Fig. 12. Ein weiterer Paper Prototyp

ist ebenfalls der Greifarm skizziert, ebenso die vorhandenen Sensoren. Neben der Kamera vorne, kamen noch Beleuchtungselemente hinzu. Des Weiteren wurde im Use-Case festgehalten, dass der Rescue Roboter, durch einen Operator mit der zu rettenden Person kommunizieren muss. Dafür wurde ein Lautsprecher im vorderen Bereich eingezeichnet.

[Alexander Wilms]

E. Iterative Designentwicklung

Im weiteren Verlauf wurden Veränderungen, Ergänzungen und Erneuerungen schriftlich festgehalten. Es gab mehrere Feedback Runden und Gespräche, um das Design immer weiter zu verfeinern. Der zweite Prototyp wurde in SolidWorks erstellt. An diesem wurden sukzessiv Änderungen vorgenommen. Dabei wurde erst der grobe Body erstellt. Es folgten im Verlauf neue Bauteile, die der Gruppe hinzugefügt wurden. Dabei wurden viele Bauteile selbst erstellt und positioniert. Andere Bauteile, wie zum Beispiel ein komplexer Kran, wurde zur Veranschaulichung als Bauteil, am Roboter eingebettet.

[Alexander Wilms]

F. Finales Rendering des Prototypen

Bis hin zum fertigen Modell des Rescue Roboters und Renderings gab es viele iterative Designscheidungen. Dabei wurden teils in gemeinsamen Streaming Sessions Bauteile erstellt und passend am Modell ergänzt. Die weiteren Designscheidungen leiteten sich aus dem Use-Case und den Requirements ab. Zudem erfolgte eine Hardware und Bauteil Analyse. Aus dieser entwickelten sich bestimmte Bauteile und

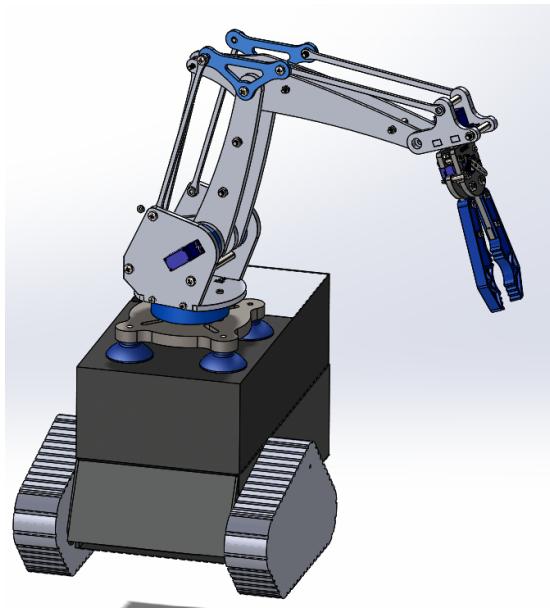


Fig. 13. Grundbau des Prototypen in SolidWorks



Fig. 14. Finales Rendering des Prototypen

Funktionen die am Roboter hinzugefügt wurden. Dieser Entwurf des Rescue Roboters zeigt die wichtigsten Bestandteile und greift die Konzeptideen mit auf. Diese werden visuell und anschaulich in einem 3D-Rendering dargestellt. Die Bauteile und Relationen bzw. Proportionen sind nicht maßstabsgetreu. Bis hin zur finalen Umsetzung und Realisierung des kompletten Projekts müssten diese noch verfeinert werden, um eine reale Funktion sicher und korrekt zu gewährleisten.

[Alexander Wilms]

VII. EVALUATION

A. Klassendiagramm

Die Zusammenhänge des gesamten Programs werden durch das in Abbildung 15 gezeigte Klassen Diagramm abgebildet. Zu sehen ist in der Mitte der Roboter. Diese Klasse

repräsentiert den Roboter mitsamt der Logik. Hierdrin werden auch die Objekte der Sensoren angelegt. Die Sensoren stellen eine Klasse dar, diese vererbt eigenschaften an die einzelnen Implementierungen der Sensoren. Für jeden Typ Sensor der in dem Roboter eingebaut wurde ist eine Klasse angelegt. Die einzelnen Implementierungen werden später ausgiebiger beschrieben. In der Fahrzeug Klasse wird weiterhin der Antriebsstrang angelegt. In dem Antriebsstrang sind die einzelnen zur Fortbewegung notwendigen Motoren angelegt. Die Umgebung repräsentiert unter anderem die Karte und die Referenzstationen.

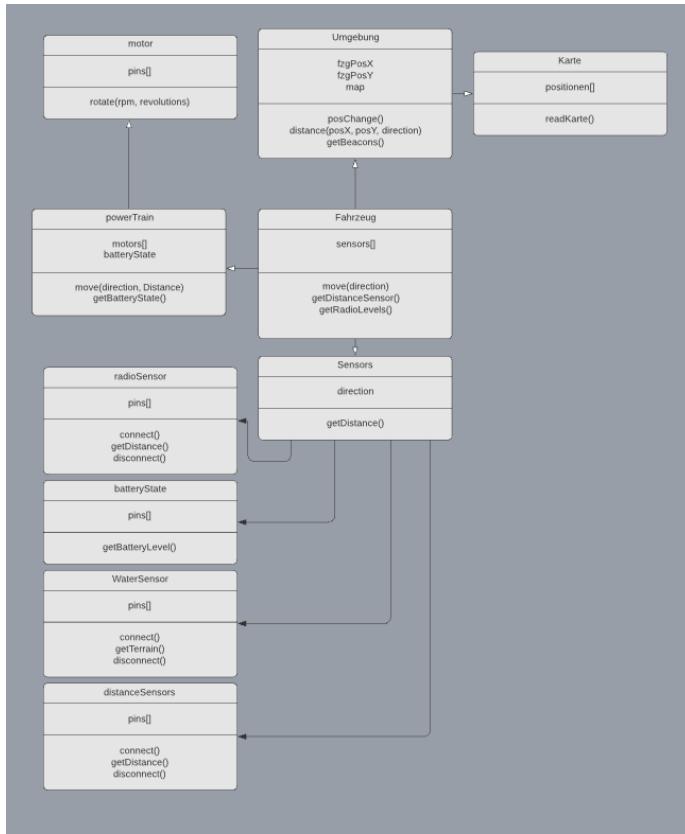


Fig. 15. Klassendiagramm

[Domenic Drechsel]

B. Karten Generierung

Gegeben wurde die Karte auf der der Roboter navigieren soll. Diese ist in Abbildung 16 zu sehen. Damit die Simulation möglichst nahe an der Realität bleibt wurde die Karte mit der maximalen Auflösung eingelesen und in ein zwei-Dimensionales Array umgewandelt. Zur Vorbereitung für die Umwandlung sind die markanten Kartenpunkte in verschiedenen Farben eingezeichnet worden. Die genutzten Farben mussten sich von den anderen markant unterscheiden. Der im RGB Farbsystem als R0 G255 B0 gekennzeichnete Wert ist ein Reiner Grünton. Dieser ist nirgendwo in dem Bild sonst zu finden. Der Zweite Farbton ist ein eindeutiges Pink

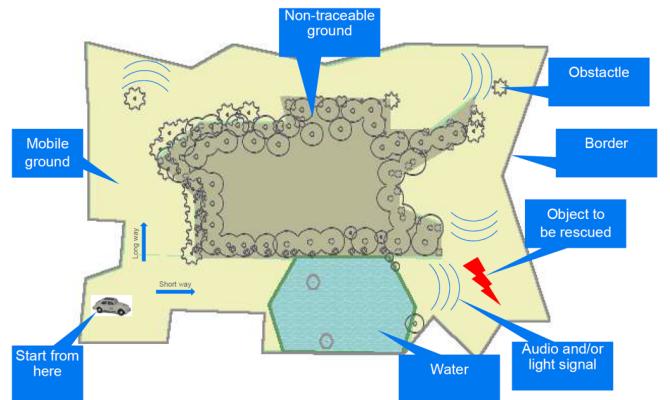


Fig. 16. Karte

mit den Farbwerten R255 G0 B230. Auch diese Farbe ist nirgendwo auf dem Bild zu finden. Die Grünen Bereiche können von dem Roboter nicht überfahren werden. Diese Stellen Hindernisse und Mauern dar. Wenn der Roboter dennoch dadurch fahren möchte muss dieser das 'Hindernis' entfernen. Dies Geschieht mit dem angebrachten Greifarm. Die Pinken Bereiche kennzeichnen einen Bereich in dem Wasser zu finden ist. Dieser ist Relevant damit wir die verschiedenen Untergründe Simulieren können auf denen der Roboter unterwegs ist. Die finale Einfärbung ist in Abbildung 17 zu sehen. Das Bild wird

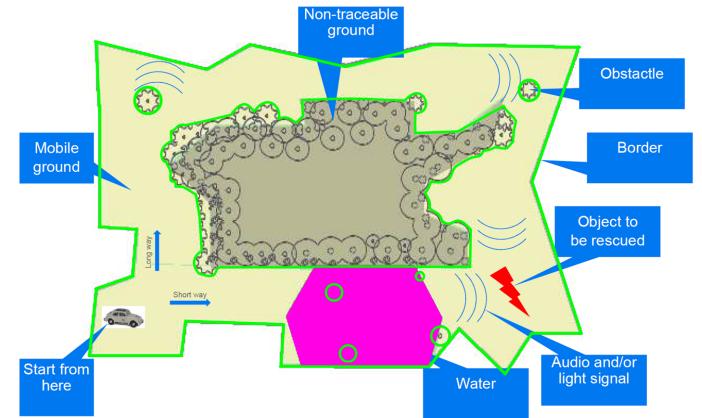


Fig. 17. Karte bearbeitet

Pixel für Pixel der Reihe nach eingelesen, Die Farbe des Pixels wird mit den genannten Farben abgeglichen. Wenn der Pixel keine der definierten Farben hat ist dies ein Pixel, der befahrbar ist. Aus den Daten wird ein 2D Integer Array erzeugt. Eine Null wird eingetragen, wenn der Pixel befahrbar ist. Wenn ein grün gekennzeichnetes Hinderniss auf dem Pixel ist wird dafür eine eins eingetragen. Bei Wasser als Untergrund wird eine Zwei in das Array eingefügt. Das entstandene Array wird in einem CSV Dateiformat abgelegt. Das CSV Format hat den großen Vorteil, dass jeder Eintrag durch ein Semikolon von den umliegenden Einträgen getrennt ist. Somit ist diese Datei simpel einzulesen.

In der Aufgabe angegeben waren auch Referenzstationen. Diese Stationen kennen Ihre eigene Position auf der Karte und können diese an den Roboter senden. Vom Roboter Empfangen werden diese Signale mit den Radio Empfängern. Hinterlegt sind die Genaugen Positionen sowie die Namen der Referenzpositionen in einer CSV Datei. Diese wird beim generieren der Karte geladen und die Stationen werden in einer Liste hinterlegt auf die der Radio Empfänger zugreifen kann. Die Referenzstationen sind auf der Karte oben Links und oben Rechts angelegt.

[Domenic Drechsel]

C. Nutzung in der Simulation

In der Simulation wird die generierte Datei eingelesen und in einem Array abgelegt. Die Roboterlogik darf aufgrund der Hierarchie nicht direkt mit der Karte in Beziehung stehen. Jedwegliche Erfassung der Umgebung findet nur über die Sensoren statt. Damit diese Karte genutzt werden kann sind mehrere Methoden in der Klasse hinterlegt die das Aktuelle Terrain zurück geben und die Entfernung in eine der Vier Himmelsrichtungen. In dem ersten Ansatz ist es dem Fahrzeug nur möglich in Richtung der Vier Himmelsrichtungen zu fahren. Für die Zukunft vorgesehen war jedoch auch in den zwischen Himmelsrichtungen, also Nord Osten, Süd Osten, etc..., zu fahren. Daher werden die Himmelsrichtungen intern wie in Abbildung 18 zu sehen benannt. Norden ist als Richtung

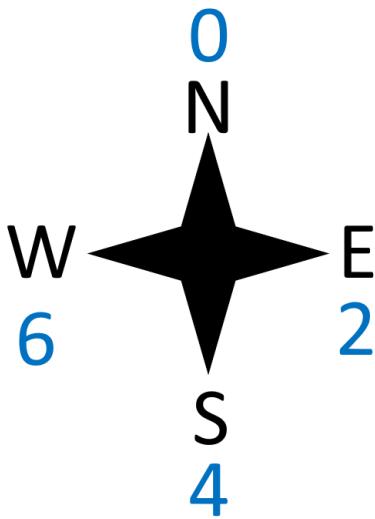


Fig. 18. Zuordnung Himmelrichtung, Zahlen

0 gekennzeichnet. Darauf aufbauend wird im Uhrzeigersinn hochgezählt. Westen ist intern als Richtung Sechs benannt.

[Domenic Drechsel]

D. Sensorik

Die Sensorik wird durch eine dafür vorgesehene Klasse dargestellt. Es sind verschiedene Typen von Sensoren an dem Fahrzeug angebracht. Die Gemeinsamkeiten der Sensoren

wurden in die Sensorik Klasse ausgelagert. Jeder Sensortyp erbt die ausgelagerten Eigenschaften von dieser Klasse. Die Sensorik Klasse ist in Abbildung ?? zu sehen

```
class sensor{
    6 references
    public int[] Pins;
    5 references
    public String Name;

    4 references
    public void connect(){
        Console.WriteLine(
            "Verbindung aufgebaut");
    }
    0 references
    public void disconnect(){
        Console.WriteLine(
            "Verbindung geschlossen");
    }
}
```

Fig. 19. Sensorik Klasse

Gemeinsamkeiten der Sensoren sind die Pins. Jeder Sensor hat eine definierte Anzahl an Pins mit denen er an das Fahrzeug angeschlossen ist. Eine weitere Gemeinsamkeit ist das Öffnen und schließen von Verbindungen. Dafür sind die Connect und disconnect Methoden vorgesehen. Da dies lediglich eine Simulation ist wird in diesen Methoden keine Physische Verbindung zu den Sensoren aufgebaut. Es wird lediglich in der Konsole ausgegeben, dass die Sensoren verbunden sind. In der Realität müssten hier Fehlerbehandlungsrouterien implementiert werden die eventuell auftretende Kommunikationsfehler erkennen. Davon wird im Rahmen der Simulation jedoch abgesehen. Folgende Sensortypen werden am Fahrzeug angebaut werden.

- Wasserstand Sensor
- Distanz Sensoren
- Radio Sensor

Jeder der Sensoren hat zusätzlich zu den vererbten Eigenschaften noch die Methode um die jeweilig generierten Sensordaten auszulesen. In Abbildung 20 ist die Implementierung des Wasser Sensors zu sehen. Wie in dem Konstruktor zu sehen ist, werden die geerbten Eigenschaften 'Name' und 'Pins' mit an die Klasse übergebenen Werten gefüllt. Nachdem die Relevanten Werte eingetragen sind wird die connect Methode ausgeführt. Wie bereits beschrieben baut diese die Verbindung zu dem jeweiligen Sensor auf. Die Methode 'isWater' wird genutzt um an der gegebenen Position abzufragen ob das Fahrzeug im Wasser steht oder nicht. Bei jedem Schritt, den der Roboter zwischen Start und Endpunkt macht werden die Sensoren abgefragt. Wenn der Distanz Sensor in die gegebene Fahrtrichtung eine Distanz von nur einem Block ausgibt hält der Roboter an und entfernt das Hindernis. Diese Methode

```

class waterSensor:sensor{
    1 reference
    public waterSensor(String _Name, int _pinOne, int _pinTwo){
        Name = _Name;
        Pins = new int[]{_pinOne, _pinTwo};
        connect();
    }
    2 references
    public string isWater(int posX, int posY){
        string terrain = Karte.getTerrain(posX, posY);
        return terrain;
    }
}

```

Fig. 20. Wassersensor Klasse

zum Hindernis Entfernen ist wie das verbinden der Sensoren in dieser Simulation lediglich eine Konsolen Ausgabe. Im Anschluss an die Hinderniss Erkennung findet eine Erkennung statt, auf welchem Terrain der Roboter unterwegs ist. Dazu wird der Wasser Sensor abgefragt. Wenn dieser auf der Karte an der Position des Roboters Wasser ausliest wird der Antrieb von Land auf Wasser umgestellt. Damit diese Umstellung visualisiert werden kann existieren die Klassen 'Motor' und

In Abbildung 21 ist der Programm ablauf plan für das Teilprogramm zu sehen.

[Domenic Drechsel]

E. Robot

Die Klasse Robot stellt den Rescue Robot dar. Zu Beginn verbindet sich der Robot in der **Methode makeSensors()** mit den acht Abstandssensoren, dem Wassersensor und dem Radiosensor. Dies ist auf Figur 22 zu sehen. Dabei wird auch die dazugehörige Pinbelegung deklariert.

Für die Abstandssensoren wird eine Liste erstellt, zu der die acht Sensoren, inklusive Name, Pinbelegung und Richtung, hinzugefügt werden.

Figur 23 zeigt die **Methode drive()**. In der Methode drive() wird ein Zufallswert zwischen 0 und 2 generiert, welcher dann in einer If-else-Anweisung über den zu fahrenden Weg des Rescue Robots entscheidet. Ist der zufällig generierte Wert null, so fährt der Rescue Robot den kurzen Weg, sonst fährt er den langen Weg.

In jedem Fall greift die Methode drive() auf die Liste der eingelesenen Referenzpunkte zurück, welche der Rescue Robot abfährt.

Die **Methode fahreVonBis()**, zu sehen auf Figur 24, greift auf die Methode getWay() in der Datei RobotLogic (siehe VII-F) zu, um den zu fahrenden Weg von einem Signalposten zum Anderen mit Rücksicht auf die Hindernisse zu berechnen. Diese Methode fahreVonBis() führt mithilfe eines Additionszuweisungsoperator (siehe Figur 24 gesamtWeg += p4.distance;) dazu, dass der gesamte Weg abgefahren wird. Zudem wird auch übergeben, was der zu befahrende Untergrund ist (siehe Figur 24 power.drive(startpos[0], startpos[1], p4.direction, p4.distance, water);).

Der zu fahrende Weg wird in zwei zusätzlichen Methoden

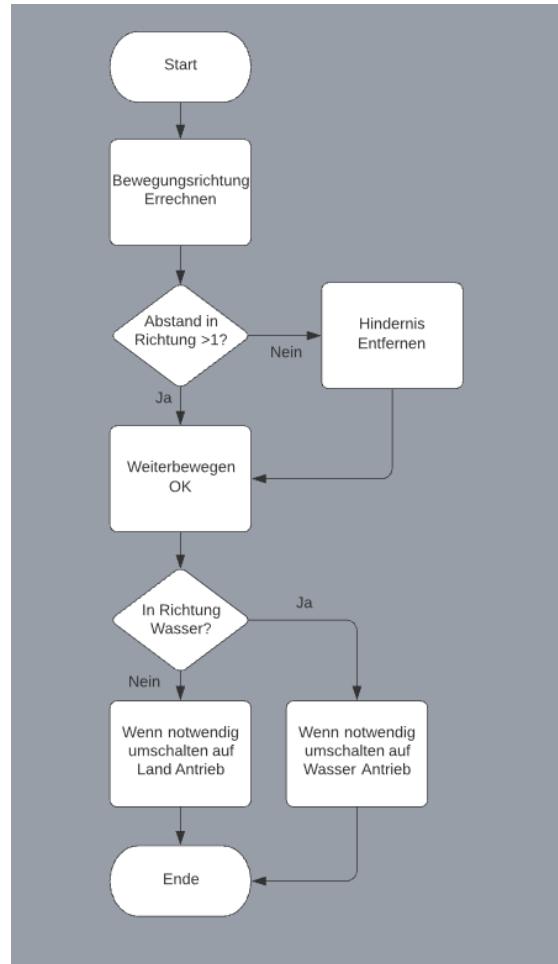


Fig. 21. Programmablauf Plan Terrain

```

public void makeSensors()
{
    Console.WriteLine("---- Sensoren Verbinden ----\n");
    sensorik.Add(new distanceSensor("Nord", 1, 2, 0));
    sensorik.Add(new distanceSensor("Nordost", 3, 4, 1));
    sensorik.Add(new distanceSensor("Ost", 5, 6, 2));
    sensorik.Add(new distanceSensor("Sudost", 7, 8, 3));
    sensorik.Add(new distanceSensor("Sud", 9, 10, 4));
    sensorik.Add(new distanceSensor("Suidwest", 11, 12, 5));
    sensorik.Add(new distanceSensor("West", 13, 14, 6));
    sensorik.Add(new distanceSensor("Nordwest", 15, 16, 7));
}

foreach (distanceSensor sensor in sensorik)
{
    Console.WriteLine("Sensor:" + sensor.Name + ";" + sensor.Pins[0] + ";" + sensor.Pins[1] + ";" + sensor.viewDirection);
}
water = new waterSensor("Wasser Sensor", 17, 18);
radioPlace = new radioSensor("Radio Sensor", 19, 20);

Console.WriteLine("---- Sensoren wurden verbunden ----\n");
}

```

Fig. 22. Methode makeSensors()

definiert, da es zwei mögliche Wege gibt. Der kurze Weg ist in der **Methode shortWay()** deklariert und der lange Weg ist in der **Methode longWay()** deklariert. Der Aufbau der beiden Methoden ist vom Prinzip gleich, in der Methode longWay() werden nur mehrere Sinalposten abgefahrene als in der Methode shortWay().

Die genannten Methoden greifen auf die vorherige erläuterte Methode fahreVonBis() zu. Figur 25 zeigt die Methode shortWay(). Der Robot fährt zuerst von dem dritten Referenzpunkt, welcher dem Startpunkt entspricht, zum zweiten Referen-

```

public void drive()
{
    radioPlace.getRefPoint();
    List<referenzPunkte> refPunkte = radioPlace.getRefPoint();
    Random rnd = new Random();
    int wert = rnd.Next(0, 2);

    if(wert == 0)
    {
        Console.WriteLine("---- Fahre den kurzen Weg ----");
        shortWay(refPunkte);
        Console.WriteLine("Person erfolgreich geborgen und am Startpunkt zurück!");
        Console.WriteLine("\n---- Mission erfolgreich beendet ----");
    }
    else
    {
        Console.WriteLine("---- Fahre den langen Weg ----");
        longWay(refPunkte);
        Console.WriteLine("Person erfolgreich geborgen und am Startpunkt zurück!");
        Console.WriteLine("\n---- Mission erfolgreich beendet ----");
    }
}

```

Fig. 23. Methode drive()

```

bool fahreVonBis(int[] startpos, int[] endPos, List<distanceSensor> sensorik)
{
    Console.WriteLine("");
    wayPoint p4 = logic.getWay(startpos, endPos, sensorik);
    gesamtWeg += p4.distance;
    power.drive(startpos[0], startpos[1], p4.direction, p4.distance, water);
    return true;
}

```

Fig. 24. Methode fahreVonBis()

zpunkt, welcher der zu bergenden Person entspricht. Daraufhin wird die Person identifiziert und die Livesteuerung wird aktiviert. Nachdem die Person erfolgreich per Livesteuerung des Roboterarms in dem Rescue Robot ist, fährt der Robot zurück von dem zweiten Referenzpunkt (geborgene Person) zum dritten Referenzpunkt (Startpunkt). Da die Referenzpunkte aus einem X- und einem Y-Wert bestehen, sind sie als Arrays definiert. Zum Schluss gibt die Methode noch den in der Methode fahreVonBis() errechneten zurückgelegten Gesamtweg aus.

```

public void shortWay(List<referenzPunkte> refPunkte)
{
    Console.WriteLine("");
    fahreVonBis(new int[] { refPunkte[3].x, refPunkte[3].y }, new int[] { refPunkte[2].x, refPunkte[2].y }, sensorik);
    Console.WriteLine("Bei Person angekommen. Person wurde als Albert identifiziert.\nSteuerung wird an Hauptquartier übergeben (Live Steuerung.)");
    fahreVonBis(new int[] { refPunkte[2].x, refPunkte[2].y }, new int[] { refPunkte[3].x, refPunkte[3].y }, sensorik);
    Console.WriteLine("Innentag: (# blocks : gesamtWeg");
}

```

Fig. 25. Methode shortWay()

[Karin Glöwing]

F. Robot Logik

Die Datei robotLogic besteht aus einer **Klasse wayPoint** und einer **Klasse Logic**.

Die **Klasse wayPoint** (siehe Figur 26) definiert eine Methode wayPoint() mit den Variablen der Richtung und der Distanz. Zudem werden die Variablen für die Erreichbarkeit des nächsten Signals und der Hindernisse deklariert.

Die eigentliche Logik des Robots ist in der **Klasse Logic** definiert.

Figur 27 zeigt die **Methode fahren()**, welche das Array move mit dem Inhalt der Richtung und der Distanz, die der Robot fahren soll, enthält.

In der **Methode calcDirDist()** (siehe Figur 28) wird die Richtung, in die der Rescue Robot fahren soll und die Distanz, die der Robot zurücklegen soll, jeweils für die X- und Y-Koordinaten, ermittelt.

```

class wayPoint
{
    public int direction;
    public int distance;
    public bool reachable = false;
    public int hindernis;
    OVerweise
    public wayPoint(int _dir, int _dist)
    {
        direction = _dir;
        distance = _dist;
    }
    I-Verweis
    public wayPoint(){}
}

```

Fig. 26. Methode wayPoint()

```

public void fahren(int _direction, int _distance)
{
    move = new int[] { _direction, _distance };
}

```

Fig. 27. Methode fahren()

Die Variablen deltaX und deltaY entsprechen den Werten der zu fahrenden Distanz in X- und Y-Richtung. Mit einer If-else-if-Anweisung wird jeweils für den X- und den Y-Wert festgelegt, in welche Richtung der Robot welche Distanz zurücklegen soll. Mit den Zuweisungen point.direction=direction und point.distance=distance wird festgelegt, dass die If-else-if-Anweisungen immer für die aktuelle Position ausgeführt wird. Dafür wird point als aktuelle Position aus der Methode wayPoint (siehe Figur 26) verwendet.

```

wayPoint calcDirDist(int[] startpos, int[] endPos, List<distanceSensor> sensorik)
{
    wayPoint point = new wayPoint();

    int deltaX = startpos[0] - endPos[0];
    int deltaY = startpos[1] - endPos[1];
    int direction = -1;
    int distance = -1;

    //X
    if (deltaX > 5)
    {
        direction = 6;
        distance = deltaX;
    }
    else if (deltaX < -5)
    {
        direction = 2;
        distance = deltaX * -1;
    }

    //Y
    if (deltaY > 5)
    {
        direction = 0;
        distance = deltaY;
    }
    else if (deltaY < -5)
    {
        direction = 4;
        distance = deltaY * -1;
    }

    point.direction = direction;
    point.distance = distance;

    return point;
}

```

Fig. 28. Methode calcDirDist()

Figur 29 zeigt die **Methode getWay()**, welche den Abstand zum nächstliegenden Objekt, sei es eine Wand oder ein Hindernis, ermittelt.

Diese Methode greift auf die Informationen der Methode calcDirDist() zu, um die Richtung und die Entfernung (Werte entsprechen der Luftlinie), in die der Abstand zum nächstliegenden Objekt berechnet werden soll, zu erfahren. Daraufhin wird der Distanzsensor aus der Datei Sensoren (beschrieben von Domenic Drechsel) abgefragt, wie weit die Distanz zum nächsten Objekt (Wand oder Hindernis) ist. Mit einer If-else-Anweisung wird ermittelt ob die Entfernung aus der Methode calcDirDist() größer ist als die Strecke bis zum

nächsten Objekt. Ist dies der Fall, so ist der von der Methode calcDirDist() ermittelte Zielpunkt nicht erreichbar und es ist ein Hindernis im Weg. Ist dies nicht der Fall, ist kein Hindernis im Weg und der Zielpunkt ist erreichbar.

```
public wayPoint getWay(int[] startpos, int[] endPos, List<distanceSensor> sensorik)
{
    wayPoint point = calcDirDist(startpos, endPos, sensorik);
    Console.WriteLine("Frage Abstandssensoren in Richtung {0}/{1}, point.direction);
    int dist = sensorik[point.direction].getDistance(startpos[0], startpos[1]);

    //Punkt erreichbar?
    if (dist > point.distance)
    {
        point.reachable = false;
        point.hindernis = dist;
    }
    else
    {
        Console.WriteLine("Der Weg vom Startpunkt {0}/{1} zum Endpunkt {2}/{3} ist in Richtung {4} mit der Entfernung {5}",
                        startpos[0], startpos[1], endPos[0], endPos[1], point.direction, point.distance);
        point.reachable = true;
    }
    return point;
}
```

Fig. 29. Methode getWay()

[Katrin Glöwing]

G. Hardware Analyse

Im Folgenden wird die Hardware und Bauteil Analyse näher beschrieben. Diese folgt zu den Designentscheidungen und ist ein wichtiger Bestandteil, um bestimmte Anforderungen zu erfüllen.

Wie im vorherigen Teil bereits erläutert gibt es nahezu unendlich viele Szenarien, in denen ein Rescue Roboter eingesetzt werden kann. Das Projekt befasst sich mit dem Szenario eines Brands, in dem eine Person gerettet werden muss. Dabei muss der Roboter in der Lage sein, Hindernisse wegzuräumen und zu passieren. Zudem muss dieser an Land fahren können und auch Wasser überqueren. Des Weiteren soll der Roboter in der Lage mit einer Person zu kommunizieren. Hierfür gibt es eine Anforderungsliste und ein bestimmtes Use-Case. Daraus geht hervor das unterschiedlichste Aufgaben zu bewältigen sind. Diese müssen nachweisbar zu belegen sein.

Bei der Konstruktion eines Konzept Roboters für den realen Gebrauch, gibt es verschiedene Bereiche. Sowohl Software-technische Bereiche als auch Hardware-technische Bereiche. Die folgende Analyse befasst sich mit den Hardware-Anforderungen. Zu Beginn der Hardware und Mechanik Analyse wurde eine Tabelle erstellt. Diese listet verschiedene Bauteile auf und zeigt die Vor- und Nachteile. Zudem gibt es eine Zusammenfassung mit der Nutzbarkeit für das oben beschriebene Use Case. Nach der Analyse folgte eine darauf basierende Entscheidung wie das Design und der Roboter konzipiert sind.

[Alexander Wilms]

H. Forttrieb

Als erster Teilbereich wurde der Forttrieb an Land untersucht. Hier wurden Ketten, normale Reifen und Mecanum Räder untersucht. Auch wenn die Ketten schwer sind, haben Sie doch einen großen Vorteil gegenüber den anderen beiden Typen. Sie sind robust und hitzebeständig. In dem Szenario gibt es ein Feuer und brennende Hindernisse. Dabei gibt es

Auflistung der Hardware & Mechanik / Vorteile & Nachteile

Hardware / Mechanik	Beschreibung	Material	Vorteile	Nachteile	Zusammenfassung
Forttrieb Ketten	Gummierter Kunststoff		Leicht, Gummig, Schwer, Metall, Montage simple, Formvari, 3D Druck	Hitze, Kälte, nicht langlebig, leicht schall, geringe Belastbarkeit	Generell gut für Steigungen und geneigt für Hitze und Brannförderung. Jedoch schlecht geeignet für Hitze und Brannförderung. Ketten und Gummirte können bei fehlender Hitze schleichen können. Dadurch kann ein Einsatz je nach Betriebszeit nicht korrekt durchgeführt werden.
Forttrieb Ketten	Metall		Hitze, Kälte, langlebig, stabil, hohe Belastbarkeit	Schwer, Teuer, Schwere nicht, Straße beschädigen, komplexe Herstellung	Ketten sind generell eher schwer und teuer. Der große Vorteil: Ein Kettenfahrzeug kann in jedem Terrain fahren. Hügel, Geröll und Erde. Auch Hindernisse und Steigungen sind leicht passierbar. Zudem sind die Metallketten für große Temperaturschwankungen geeignet. Soviel wie Hitze und Kälte und können überwunden werden. Kettenfahrzeug auch für Brannförderung eine gute Möglichkeit. Zudem kann ein Kettenfahrzeug sich auf der Stelle drehen und ist von der Fahrtrichtung flexibel.
Forttrieb Räder	Mecanum Räder (Gumm)		360° Drehung, Straße, Sand, Erde, Hügel	langsam, Wasser, Hitze, Kälte, Steine	Mecanumräder sind generell praktisch da sie sich in jede Richtung drehen können. Denkt wäre das Fahrzeug sehr flexibel und wendig. Jedoch ist die Beschichtung aus Gummie und sonst für Hitze eher ungeeignet. Des Weiteren könnte Geröll die Mechanik blockieren und somit den Einsatz frühzeitig beenden.
Forttrieb Räder	Reifen (Gumm)		Schnell, Straße, Sand, Erde, Hügel, günstig, Verfügbarkeit	nicht drehbar, Wasser, Hitze, Kälte, Steine	Räder bzw. Reifen sind fast überall verfügbar. Die Hinterhaltung bei einer Beschädigung wäre leicht und unkompliziert. Da die Reifen eine Gummibeschichtung sind sie nicht für Hitze beständig gemacht auf Grund des Gummies ist das kein Vorteil. Zudem haben Reifen nicht den hohen Grip und rutschen z.B. im Sand sterben bleiben. Auch größere Hindernisse können nicht überwunden werden.

Fig. 30. Auflistung der Hardware und Mechanik /Vor- und Nachteile (1)

Model	Chassis	Vollmetall	stabil, Hitze, Kälte, Explosions, Feuer, hohe Belastbarkeit	hoher Gewicht, relativ teuer	Auch wenn das Chassis etwas schwerer ausfällt und das Material enorm. Es ist gegen fast alle äußeren Einflüsse geschützt und hat auch eventuelle Explosions aus.
Model	Chassis	Kunststoff	leicht, Hitze, Herstellung einfach, gummig	instabil, Hitze, Kälte, Explosions, Feuer, keine Belastbarkeit	Die Herstellung ist zwar einfach und das Chassis wäre leicht jedoch ist die Gefahr dass es durch Hitze oder Einschläge zerstört wird.
Model	Chassis	Super Materialien	leicht, stabil, extreme Hitze, extreme Kälte, Explosions, Feuer, hohe Belastbarkeit	Anschaffung schwer, teuer, Herstellung, Forschung	Supermaterialien sind zwar in der Herstellung teuer und schwer zu beschaffen aber die Vorteile sind immens. Sie sind leicht, stabil und extrem hitzebeständig. Somit wären sie für risikante und gesonderte Situation ideal geeignet.
Rettung	Rettungskran / Kran	Vollmetall	Hitze, Kälte, langlebig, stabil, hohe Belastbarkeit	Schwer, Teuer	Der Vorteil beim Vollmetall ist, dass er sehr stabil ist und sehr lange genutzt werden kann ohne ihn auszubauen. Zudem kann man auch bei unterschiedlichen Wetter Bedingungen damit arbeiten. Ein Nachteil ist, dass man bei der Rettung des Patienten vorsichtig sein muss ihn nicht zu verletzen.
Rettung	Rettungskran / Kran	Kunststoff	leicht, günstig, 3D Druck, einfache Herstellung	Hitze, Kälte, instabil, geringe Belastbarkeit	ist instabil und nicht nutzbar im richtigen Betrieb, nutzbar um eine miniatur Darstellung zu erstellen da es 3D Druckbar ist.
Rettung	Rettungskran / Kran	Super Materialien	leicht, stabil, extreme Hitze, extreme Kälte, Explosions, Feuer, hohe Belastbarkeit	Anschaffung schwer, teuer, Herstellung, Forschung	Supermaterialien sind vielleicht amwendbar in Feuer/Kälte-Situationen könnte der Kran auch hitzebeschützt sein so das der Patient nicht vom heißen Stahl verletzt werden kann.
Antrieb	E-Batterie/Akku	Lithium	Elektro, aufzuladen	Allzweck, Teuer in der Anschaffung	Sind wiederanlaufbar und besser für die Umwelt, doch haben sie so viel Leistung wie ein Brennstoffmotor
Antrieb	Brennstoffmotor	Benzin/Diesel	günstig, universell	explosionsgefähr, Leck und Auslauftaupar	Haben eine hohe Leistung, doch sind schlecht für die Umwelt, könnte zu Problemen führen bei einer Explosion.

Fig. 31. Auflistung der Hardware und Mechanik /Vor- und Nachteile (2)

eine hohe Temperaturentwicklung. Für den Fall, dass gummierte Reifen eingesetzt werden, könnten diese schmelzen. Wenn dies eintritt verlieren Reifen ihre Funktionalität und ein korrekter Einsatz muss gegebenenfalls frühzeitig beendet werden.

Somit ist die Auswahl auf robuste Ketten gefallen. Ein weiterer Vorteil ist die Steigung und Passierbarkeit von Hindernissen. Durch einen Kettenantrieb ist es möglich über

Geröll und verschiedene Bodentypen zu fahren, ohne sich festzufahren. Wenn der Roboter auch durch einen See fahren soll, hat der Roboter mit Kettenantrieb, an dem Ufer mehr Grip und kann besser hinein- und herausfahren.

[Alexander Wilms]

I. Chassis

An zweiter Stelle wurde das Chassis und dazu gehörige Material genauer betrachtet. Das Gehäuse und die Beschaffenheit hängen stark mit den Anforderungen zusammen. Zwar ist ein Roboter aus 3D gedrucktem Material leichter und schnell gefertigt, doch so gibt es auch hier einen wichtigen Nachteil. Der Schmelzpunkt von 3D Drucken und üblichen Kunststoffen ist recht gering. Im Falle eines Brandes könnte das Gehäuse schmelzen und das Fahrzeug in sich kollabieren.

Metall wäre besser geeignet ist aber nicht auch nicht optimal. Das hohe Gewicht wirkt sich auf den gesamten Roboter aus. Da der Rescue Roboter schwimmen soll, ist das Gewicht relativ gering zu halten, wenn möglich. Das Gehäuse könnte aus einem Supermaterial gefertigt werden, wie in der Raketenforschung. Dort gibt es ein leichtes Material welches als Hitzeschutz benutzt wird aber dennoch leicht ist. Wie zum Beispiel Kohlefaser-Verbundstoff mit einer Keramikbeschichtung. Diese Materialien sind zwar teuer in der Herstellung und Produktion aber für einen speziell konstruierten Rescue Roboter von großem Vorteil.

[Alexander Wilms]

J. Rettungsvorgang

Als nächstes wurde der Rettungsvorgang analysiert. Hierfür eignet sich ein multifunktionaler Greifarm am besten. In dem Szenario sollen Hindernisse weggeräumt werden. Zudem sollen Personen gerettet und versorgt werden. Mit einem ferngesteuerten Roboter-Greifarm der sich in mehrere Richtungen drehen und ausfahren kann, könnte dies realisiert werden. Das Material sollte auch hier wieder ähnlich speziell gewählt sein wie bei dem Chassis wegen der hohen Hitze. Um den Greifarm in mehrere Richtungen zu drehen und auszufahren werden drei unabhängige Motoren verwendet, um die geeignete Leistung aufzubringen. Für den Rettungsvorgang ist eine nach oben öffnende Doppeltür-Klappe vorgesehen. Diese kann sich öffnen, um zum Beispiel Verbandskästen mithilfe des Krans herauszugeben oder eine zu rettende Person, dort im Innenraum aus der Gefahrensituation zu befördern.

[Alexander Wilms]

K. Energieversorgung und Motoren

Die Energie Versorgung soll über E-Akkus erfolgen. Die Motoren agieren ebenfalls über elektrische Leistung. Die benutzten Elektro-Motoren und Akkus sind speziell beschichtet, um der Hitze Stand zu halten.

Es werden kein Benzin/Diesel oder Öl-Tanks und Motoren verwendet da diese das Risiko haben bei hoher Hitze zu explodieren. Zudem dürften diese kein Leck haben. Durch

austretendes Benzin könnten Brandpfützen entstehen. Des Weiteren könnte das Wasser verunreinigt werden.

[Alexander Wilms]

L. Forttrieb im Wasser

Um den Forttrieb im Wasser zu gewährleisten werden mehrere Aspekte berücksichtigt. Der Rescue Roboter verfügt über eine hinten angebrachte Schiffschraube, auf einer höhenverstellbaren Schiene. Mit Hilfe dieser, kann die Schiffschraube auf die gewünschte Höhe gebracht werden, so wie auf- und abgelassen werden, je nach Umgebung. Damit kann der Roboter sich im Wasser vorwärtsbewegen.

Um einen generellen Auftrieb zu geben, verfügt der Roboter über einen großen Luft-Hohlraum. Dieser fungiert ähnlich wie bei einem Schiff und gibt dem Roboter die Möglichkeit besser zu schwimmen. Für den weiteren Auftrieb nach oben sorgt eine leistungsstarke Pumpe die mittels Düsen, Wasser einsaugt und nach unten ausströmt. Des Weiteren befindet sich unten am Roboter ein Feuchtigkeitssensor, um zwischen Land- und Wasser zu wechseln.

[Alexander Wilms]

M. Raumaufteilung

Der Roboter besteht aus drei Hauträumen. Der oberste dient zur Personenrettung und verfügt über eine Türklappe. In diesem Raum ist die Person vor äußeren Einflüssen geschützt. Zudem befindet sich dort ein Erste-Hilfe-Kasten. Der mittlere Raum ist der Technik-Raum bzw. Motoren-Raum, welcher über eine hintere Tür für Wartungsarbeiten zugänglich ist. Der unterste Raum ist als Hohlraum für den Wasserauftrieb konzipiert.

[Alexander Wilms]

N. Technische Komponenten

Da der Rescue Roboter noch weitere Anforderungen besitzt, werden diese im Weiteren erläutert. Der Roboter muss mit dem Umweltsystem interagieren und kommunizieren können. Dafür gibt es mehrere Möglichkeiten dies zu realisieren. Verschiedene Technikkomponenten kommen zum Einsatz.

[Alexander Wilms]

O. Orientierung

Um mit der Einsatzleitung zu kommunizieren und generelle Signale senden und empfangen zu können ist eine Antenne angebracht. Zur generellen Orientierung gibt es ein GPS mit integriertem Kompass. Für die lokale Orientierung von Hindernissen und Wegen sind acht Abstandssensoren am Roboter befestigt. Diese blicken in verschiedene Richtungen und scannen die Umgebung. Die Daten werden über die Software analysiert und berücksichtigt.

[Alexander Wilms]

P. Display und LED

Für eine verbesserte Sicht bei eingeschränkten oder dunklen Begebenheiten sind vorne drei LED-Scheinwerfer angebracht. Damit der Roboter und der Operator mit der zu rettenden Person kommunizieren können ist vorne ein großes Display angebracht. Über dieses lassen sich Instruktionen und Video Chats anzeigen.

[Alexander Wilms]

Q. Multimedialglobe und Kameras

Damit der Operator noch besser mit der Person kommunizieren kann, ist ein Multimedia-Globe vorne im Greifarm eingebaut. In diesem Globe befindet sich ein Mikrofon, ein Lautsprecher und eine Kamera. So kann der Greifarm nah an die Person heranfahren und mit ihr audio-visuell den Kontakt aufnehmen.

Vorne am Roboter ist ein spezielles Kamera Array angebracht. Dieses verfügt über verschiedene Kameras und Techniken. So gibt es eine digitale Kamera für die generelle Bildübertragung. Zudem gibt es eine Thermalkamera und Infrarotkamera, welche die Temperaturgegebenheiten anzeigen können. Des Weiteren gibt es eine Nachtsichtkamera, um auch in dunklen Bereichen ein visuelles Bild zu übertragen.

Diese Techniken werden von verschiedenen Einsatzkräften bei Militär und Feuerwehr eingesetzt. Dabei werden verschiedene Systeme benutzt um mögliche Szenarien abzudecken. Die Vielseitigkeit erlaubt es zwischen unterschiedlichen Sehbereichen zu wechseln, um das Ziel zu erreichen.

[Alexander Wilms]

R. Reale Umsetzung

Die Hardware und Bauteilanalyse wurden für Designentscheidungen berücksichtigt und umgesetzt. Der Rescue Roboter verfügt über die in den Anforderungen beschriebenen Bestandteile. Da es sich um ein Konzept Entwurf handelt ist die grobe Richtung eingeleitet. Bis hin zur finalen Realisierung muss der Entwurf iterativ verfeinert werden. Der Rescue Roboter ist in der Planung ein Unikat und als Projekt zu verstehen. Für das Konstruieren werden spezielle Materialien und Sensoriken benötigt. Diese müssen teils extra gefertigt und bestellt werden. Da es sich um einen Roboter für Extremsituationen handelt in dem Menschen gerettet werden, ist die Planung enorm wichtig und muss mit mehreren Gremien und Sachverständigen geplant werden. Die Anforderungen und Richtlinien sind dabei hoch und vielschichtig komplex.

[Alexander Wilms]

S. Design

Auf dem Fig.4 ist unser fertiges Rendering vom Rescue Robot zu sehen. Jedes einzelne Bauteil ist mit den zu erfüllenden Anforderungen verbunden. Diese Verbindung wird mit einem Pfeil dargestellt.

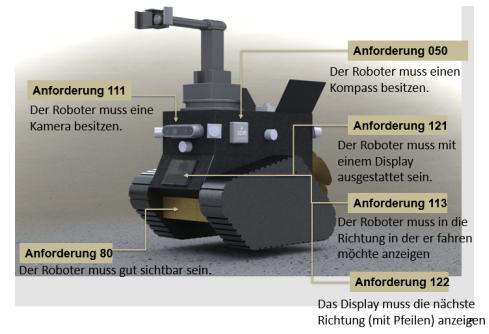


Fig. 32. Rescue Robot Vorne

T. Multifunktionskamera

(Anforderung: 111)

Hier zu sehen ist unsere Multifunktionskamera. Diese besteht aus einer Infrarotkamera, einer Thermalkamera, einer Nachtsichtkamera und einer Digitalkamera. Sie wird genutzt, um Objekte und Personen während der Fahrt in der Ferne zu erkennen.

[Justin Frommberger]

U. Led Panel

(Anforderung: 80)

Die Anforderung 80 zeigt ein dreiteiliges LED Panel, welches für die Beleuchtung genutzt wird. Zudem hilft das Licht des LED Panel dem Bediener und macht den Rescue Robot gut sichtbar für die Umgebung.

[Justin Frommberger]

V. GPS/Kompass

(Anforderung: 50)

An der Seite des Roboters wurde der Multifunktions GPS/Kompass angebracht, um die richtige Richtung und den aktuellen Standort zu ermitteln.

[Justin Frommberger]

W. Display

(Anforderung: 113/121/122)

Vorne am Robot und gut sichtbar, wurde das Display angebracht. Hier werden Nachrichten für die Umgebung dargestellt. Wie z.B das Blinken mit einem Pfeil in die Richtung, wohin er fährt. Zusätzlich kann auch über das Display, mit dem Patienten, kommuniziert werden.

[Justin Frommberger]

X. Kran mit Multimedia-Globe

(Anforderung: 70/113)

Oben, auf dem Robot, ist der bewegliche Kran befestigt. Die Hauptaufgabe des Krans ist es, den Patienten sicher zu bergen und in den Rescue Robot zu legen. Der Kran wird auch dafür verwendet um kleinere Gegenstände, die im Weg liegen, zu entfernen. Dies geschieht über die extra angefertigte

Kralle, die auch für einen sicheren Transport des Patienten sorgt. Der Kran ist ausfahrbar und kippbar um die Person auch aus schweren Situationen zu retten. Zusätzliche ist ein Multimedia-Globe befestigt. Dieser beinhaltet eine Kamera, einen Lautsprecher und ein Mikrofon, um mit der zu rettenden Person zu kommunizieren.

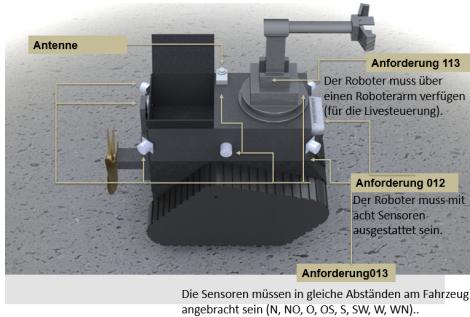


Fig. 33. Rescue Robot Oben

[Justin Frommberger]

. Wasserantrieb

(Anforderung: 031/043)

Hinten am Rescue Robot ist eine Schiene befestigt, um den daran befestigten Propeller hoch und runter fahren zu können. Der Propeller wird für den Wasserantrieb verwendet.

[Justin Frommberger]

. Ketten

(Anforderung: 044)

Zu sehen sind große stabile Ketten um auf dem Land über mehrere Hindernisse zu fahren.

[Justin Frommberger]

. Doppeltür

Die automatisch öffnenden/schließenden Türen oben am Robot sind dafür da, dass die Person sicher zurückgebracht werden kann, ohne das z.B Trümmer oder gar Feuer in das Fahrzeug gelangen kann.

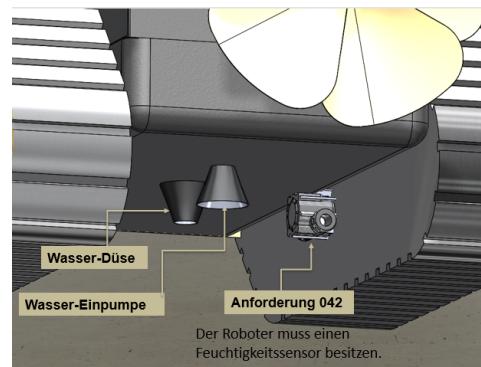


Fig. 35. Rescue Robot Unten

Y. Abstandssensoren

(Anforderung: 012/013)

In allen Himmelsrichtungen sind Abstandssensoren an den Robot angebracht. Diese sorgen dafür, dass unser Rescue Robot die Umgebung erfassen kann und Hindernisse frühzeitig erkennt.

[Justin Frommberger]

Z. Antenne

Oben auf dem Robot wurde eine Antenne befestigt um Signale zu empfangen.

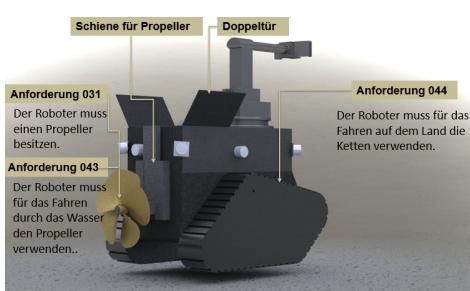


Fig. 34. Rescue Robot Hinten

[Justin Frommberger]

. Wasser-Düse

Die unten angebrachte Wasserdüse wird genutzt, um dem Rescue Robot im Wasser Auftrieb zu geben, da er durch sein Gewicht nicht schwimmen kann.

[Justin Frommberger]

. Wasser-Einpumpen

Bei der Wasser-Einpumpe wird das Wasser für die Wasserdüse eingepumpt.

[Justin Frommberger]

. Feuchtigkeitssensor

(Anforderung: 042)

Der Feuchtigkeitssensor stellt fest, ob der Roboter im Wasser ist und auf seinen Wasserantrieb wechseln muss.

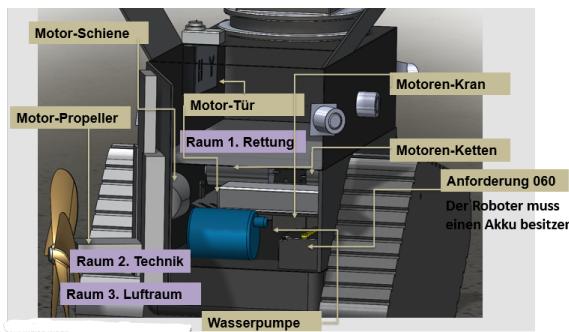


Fig. 36. Rescue Robot Offen

[Justin Frommberger]

. Raum 1

Im vorderen Raum 1 wird die Person gesichert. Im hinteren Raum 1 sind die Motoren für das öffnen und schließen der Tür angebracht.

[Justin Frommberger]

. Raum 2

Im Raum 2 sind die meisten Motoren untergebracht, wie die Motoren für die Ketten, den Kran und die Schiene. Zusätzlich ist dort eine Wasserpumpe und ein Akku untergebracht. Der Motor für den drehenden Propeller ist in der Schiene an der Tür integriert.

[Justin Frommberger]

. Raum 3

Der Raum 3 ist für den Auftrieb des Robot gedacht. Dieser ist mit Luft gefüllt und sorgt für einen sicheren Auftrieb.

[Justin Frommberger]

VIII. SUMMARY AND OUTLOOK

IX. APPENDIX

In Abbildung 37 zu sehen ist wie sich der Zeitliche Aufwand, die Commits, hinzugefügte und entfernte Zeilen sich unter den Gruppenmitgliedern aufteilen.

Name	Zeitaufwand (h)	commits	Zeilen	
			hinzugefügt	entfernt
Katrin Glöwing	85	139	2663	1731
Justin Frommberger	60	114	583	246
Alexander Wilms	50	42	454	74
Domenic Drechsel	85	55	6228	4985

Fig. 37. Zeiten Aufschlüsselung

Gesondert aufgeschlüsselt ist in Abbildung 38 ein Screenshot der Github Insights zu sehen.

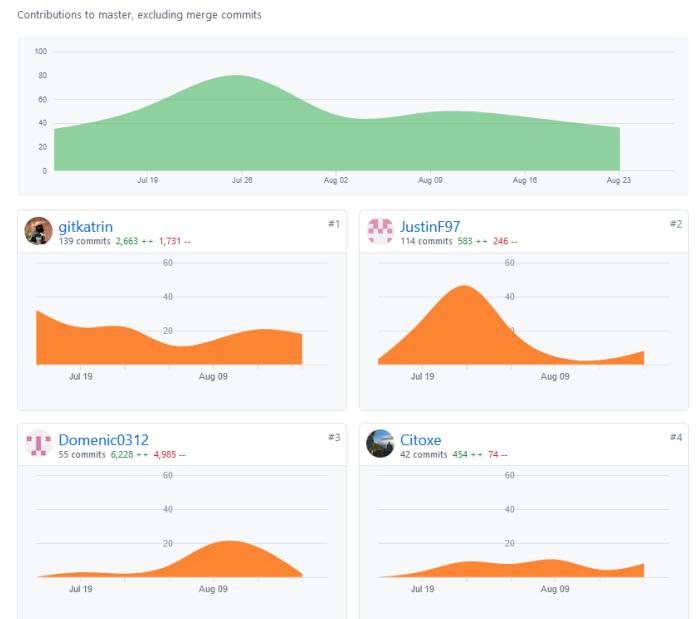


Fig. 38. Github insights

X. EIDESSTATTLICHE ERKLÄRUNG

Wir (Katrin Glöwing, Domenic Drechsel, Justin Frommberger, Alexander Wilms) erklären hiermit, dass wir das vorliegende Papier selbst verfasst und erarbeitet haben, ohne Verwendung anderer als der zitierten Quellen und Hilfsmittel. Sätze oder Teile von Sätzen, die wörtlich zitiert werden, sind als solche gekennzeichnet; andere Verweise in Bezug auf die Erklärung und den Umfang sind in den vollständigen Einzelheiten der betreffenden Veröffentlichungen angegeben. Die Ausarbeitung und Arbeit in gleicher oder ähnlicher Form wurde keiner Prüfungsstelle vorgelegt und nicht veröffentlicht. Diese Arbeit wurde noch nicht einmal teilweise in einer anderen Prüfung oder als Kursleistung verwendet.

REFERENCES

- [1] S. Robertson, and J. Robertson, "Mastering the Requirements Process: Getting Requirements Right," Pearson Education, 2012.
- [2] L. Chen and M. Ali Babar and B. Nuseibeh, "Characterizing Architecturally Significant Requirements," IEEE Software, 2nd ed., vol. 30, pp. 38–45, 2013.

IEEE conference templates contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper prior to submission to the conference. Failure to remove the template text from your paper may result in your paper not being published.