

# Die Maus

Daniel Anders, Elektrotechnik und Informationstechnik  
Otto-von-Guericke-Universität Magdeburg

**Zusammenfassung**—Der in diesem Seminar entwickelte Projektroboter „Die Maus“ befasst sich mit der Entwicklung eines autonomen Fahrzeugs, das mithilfe klassischer Bildverarbeitung einen Ausgang in einem Raum erkennt und diesen verlässt, ähnlicher einer Maus, die auf der Suche nach einem Mauseloch ist. Die Konstruktion basiert auf LEGO Mindstorms, einem EV3-Baustein als Steuereinheit und einer Webcam zur Bildaufnahme. Die Bildverarbeitung erfolgt in MATLAB unter Verwendung der ComputerVisionToolbox, wobei Kantenerkennung und Polygonerkennung zur Detektion des Ausgangs eingesetzt werden. Erste Tests mit einer Konstruktion auf vier Rädern zeigten Schwierigkeiten bei Drehbewegungen, weshalb eine optimierte Version mit drei Omniwheels entwickelt wurde. Die Maus bewegt sich gezielt auf den Ausgang zu und korrigiert ihre Position anhand einer kontinuierlichen Bildauswertung. Ein Ultraschallsensor unterstützt das System, indem er erkennt, ob die Maus den Raum verlassen hat. Erste Tests bestätigen die Funktionsfähigkeit des Systems, obwohl Herausforderungen wie falsch-positive Erkennungen Optimierungspotenzial aufzeigen.

**Schlagwörter**—Bildererkennung, ComputerVisionToolbox, EV3, MATLAB, Omniwheels, Polygonerkennung

## I. EINLEITUNG

**D**EN Autofahrern der heutigen Welt werden durch digitale Computersysteme zunehmend durch Brems-, Spurhalte- und Parkassistenten unterstützt, bis hin zum autonomen Fahren. Hierfür werden Sensoren und Kameras in Echtzeitdatenverarbeitungssysteme integriert, die nicht nur strenge Echtzeitkriterien einhalten müssen, sondern auch mit hoher Zuverlässigkeit und geringer Fehlerrate funktionieren müssen. Fortschritte im Bereich der künstlichen Intelligenz verbesserten Erkennungsrate von komplexen Objekten in Kamerasystemen, lösten allerdings traditionelle Bildverarbeitung wie Kantenerkennungen aufgrund ihrer Schnelligkeit und Einfachheit für klar definierte Probleme nicht vollständig ab [1]. Klassische Einparkassistenten setzen auf Ultraschallsensoren und Rückfahrkameras, die die Aufgabe des Einparkens zwar unterstützen, allerdings diese letztendlich dem Menschen überlassen. Aus der Idee ein Fahrzeug zu entwickeln, welches selbstständig in eine Garage einparken kann entwickelte sich eine damit verwandte Idee der "Maus", die eigenständig mittels klassischer Bildverarbeitung einen rechteckigen Ausgang in einer Wand findet und so einen Raum verlassen kann.

## II. VORBETRACHTUNGEN

### A. Welche Aufgaben soll die Maus erfüllen können?

Die Maus soll als Fahrzeug sich frei in einem Raum bewegen können und eigenständig durch Rotation um die eigene Achse sich so lange drehen, bis ein Ausgang im Blickfeld der Kamera erkannt wurde. Hierfür muss es sich mit und gegen

den Uhrzeigersinn drehen können, sowie in der Lage sein Vorwärtsbewegungen durchzuführen. Die Maus soll selbst auf den Mittelpunkt eines außen schwarz markierten Ausgangs in kleinen Schritten zufahren können und in der Lage sein, nach jedem Schritt nach Bedarf sich durch Rotation des Fahrzeugs sich eigenständig neu auszurichten. Die Maus soll zusätzlich in der Lage sein unabhängig von der Kamera zu erkennen, dass es den Raum verlässt, der die Markierungen des Ausgangs zu diesem Zeitpunkt sich nicht mehr im Blickfeld befinden.

### B. Welche Komponenten werden benötigt?

Für die Konstruktion des Fahrzeugs wurde ein umfangreicher LEGO-Mindstorms-Baukasten sowie ein EV3-Baustein zu Verfügung gestellt. Der EV3 ist dabei der zentrale Steuercomputer des Fahrzeugs, der die drei Motoren und einen Ultraschallsensor ansteuert. Ein Smartphone stellt die Webcam des Systems dar und wird mit Gummibändern an der Vorderseite des Fahrzeugs befestigt.

### C. Wie erkennt die Maus den Ausgang?

Die am Fahrzeug befestigte Webcam liefert Bilddaten, welche aufgrund des hohen Berechnungsaufwands nicht lokal auf dem Fahrzeug, sondern in einer MATLAB-Umgebung mittels der als Add-on installierten ComputerVisionToolbox verarbeitet werden. Der Ausgang befindet sich in einem rechtwinkligen Raum und muss einen starken Kontrast zu den Wänden aufweisen. In der selbstgebauten Testumgebung wurde daher weiße Wände für eine zuverlässigere Erkennung des Konturen des rechteckigen und mit schwarzem Klebeband umrandeten Ausgangs verwendet. Ein nach oben gerichteter Ultraschallsensor prüft, ob die sich die Decke in unmittelbarer Nähe befindet, um zu verhindern, dass die Maus beim Verlassen des Raums durch Drehungen erneut nach dem Ausgang sucht, sondern weiterhin vorwärts fährt um den Raum vollständig zu verlassen.

## III. FAHRZEUGKONSTRUKTION UND SOFTWAREENTWICKLUNG

### A. Konstruktion

Die erste Konstruktion des Fahrzeugs (siehe Abbildung 1) basierte auf insgesamt vier Rädern, wovon die vorderen zwei einfachen Räder durch jeweils einen großen LEGO-Mindstorms Motor angetrieben wurden und die hinteren Omniwheels dem Fahrtverlauf gefolgt sind. An dieser Konstruktion wurde bereits ein Ultraschallsensor angebracht und eine mit Gummibändern eingespannte USB-Webcam lieferte die Daten zur Bildererkennung. Das per USB ferngesteuerte Fahrzeug fuhr stabil und es konnten mit dieser ursprünglichen Konstruktion

bereits erste Erfolge bei der Bilderkennung notiert werden. Mit zunehmendem Entwicklungsverlauf wurde jedoch klar, dass vor allem die Suche nach dem Ausgang durch Drehung um die eigene Achse sich als eine Herausforderung herausstellte. Abgesehen von dem großen Drehradius erhöhte das Mitziehen der Allseitenräder in einigen Fällen die Bewegungsungenauigkeit des Fahrzeugs erheblich, weshalb schlussendlich ein neuer Ansatz gewählt wurde. Für die zweite und finale Konstruktion



Abbildung 1. Erste Konstruktion auf vier Rädern mit zwei Motoren und USB Webcam

wurden drei in Stern angeordnete kleine LEGO-Mindstorm Motoren eingesetzt (siehe Abbildung 3). Die drei Omniwheels stehen soweit mit den LEGO Bauteilen umsetzbar nahezu  $120^\circ$  zueinander versetzt und bieten weiterhin Fahrstabilität. Eine einfache Y-Anordnung der Motoren hat leider nicht gereicht, da den mittig sitzende EV3-Kern mit seiner Gewichtskraft auf das Fahrgestell einwirkt und diese leicht verbiegt, wodurch die Omniwheels nicht idealen Kontakt mit dem Untergrund haben. Um dies zu verhindern, wurde ein Dreieck zur Erhöhung der Stabilität hinzugefügt (siehe Abbildung 2). Das neue Design hat den Vorteil, dass Drehung mit und gegen den Uhrzeigersinn durch Ansteuerung der Motoren in die selbe Drehrichtung bei gleicher Leistungszufuhr möglich ist und dabei keine Räder schleifen müssen.

Die Maus ist in der Lage sich vorwärts, aber auch in zwei weitere Richtungen seitwärts zu bewegen, in dem jeweils zwei Motoren mit der selben Leistung in entgegengesetzte Drehrichtungen betrieben werden, sodass zwei Omniwheels eine translatorische Bewegung hervorrufen und das dritte Omniwheel mitgezogen wird. Der Ultraschallsensor wurde seitlich an den EV3 angebracht und ist an die Decke gerichtet. An der Vorderseite wurde ein möglichst mittig positionierte Smartphone Kamera als kabellos verbundene Webcam eingesetzt, welche mit Gummibändern an der LEGO Konstruktion befestigt wurde. Somit ist der EV3 nun die einzige mit dem Hostcomputer via Kabel verbundene Komponente.

### B. Funktionsweise und Software

Der Ablauf des in MATLAB entwickelten Programms der Maus ist in der Abbildung 7 vereinfacht dargestellt. Der EV3

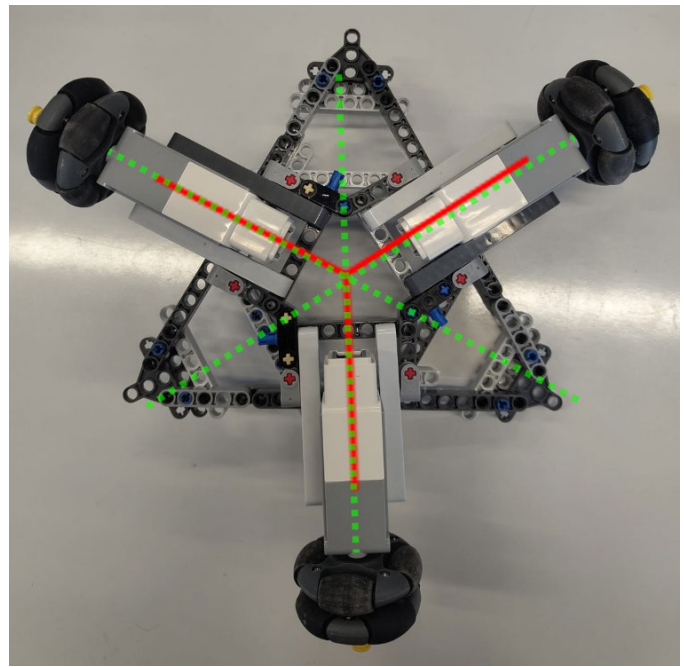


Abbildung 2. Unterseite der zweiten Konstruktion, Idealer  $120^\circ$  Versatz (rot), Verlängerte Motor-Rad-Achsen (grün gepunktet)

wurde mittels der EV3-Toolbox [2] an MATLAB angebunden und ermöglichte eine Fernsteuerung des Fahrzeugs per USB. Eine auf dem Smartphone installierte App teilte die Kamera im lokalen WLAN mit dem Hostcomputer, auf dem der geteilte Kamerastream wie eine gewöhnliche Webcam abgegriffen werden kann. In MATLAB wurde die ComputerVisionToolbox als Add-On installiert und verwendet, um sowohl ein Webcam Objekt zu erstellen, als auch an den Snapshots dieses Objektes dann die Kanten und Polygonerkennung durchzuführen. Ein Snapshot steht dabei für den aktuell verfügbaren Frame, der in MATLAB als eine  $3 \times 1280 \times 720$  große Matrix gespeichert wird.

Das RGB codierte Farbbild wird mit dem `rgb2gray()` Befehl zuerst in eine Graustufenmatrix, ähnlich der Darstellung in Abbildung 4 umgewandelt, damit anhand dieser dann die Kantenerkennung durchgeführt werden kann. Für die Kantenerkennung wurde der `edge()` Befehl mit einem Schwellwert von 0.7 sowie Canny als Erkennungsmethode verwendet [3], wobei die ermittelten Kanten des Ausgangs sich in Abbildung 5 dargestellt sind. Die tatsächliche Polygonerkennung übernimmt `regionprops()` mit der Konfiguration aus dem in dem Moment bereitgestellten schwarz-weiß Bild die Bounding Box [4] sowie Fläche der gefundenen Regionen zu ermitteln. Bereits hier können anhand dieser Daten ein Großteil der gefundenen Objekte gefiltert werden, da zu kleine oder große Rechtecke sowie Objekte mit unrealistischen Seitenverhältnis mit hoher Sicherheit verworfen werden können.

Als Ausgang interpretiert wird aus den verbleibenden Regionen das Rechteck, das nach einer Maximum-Suche den größten Flächeninhalt aufweist. Die Bounding Box dieser Fläche sowie dessen geometrischer Mittelpunkt sind in Abbildung 6 dargestellt. Sollte keine Region gefunden werden oder in Frage kommen, prüft der Ultraschallsensor, ob sich eine

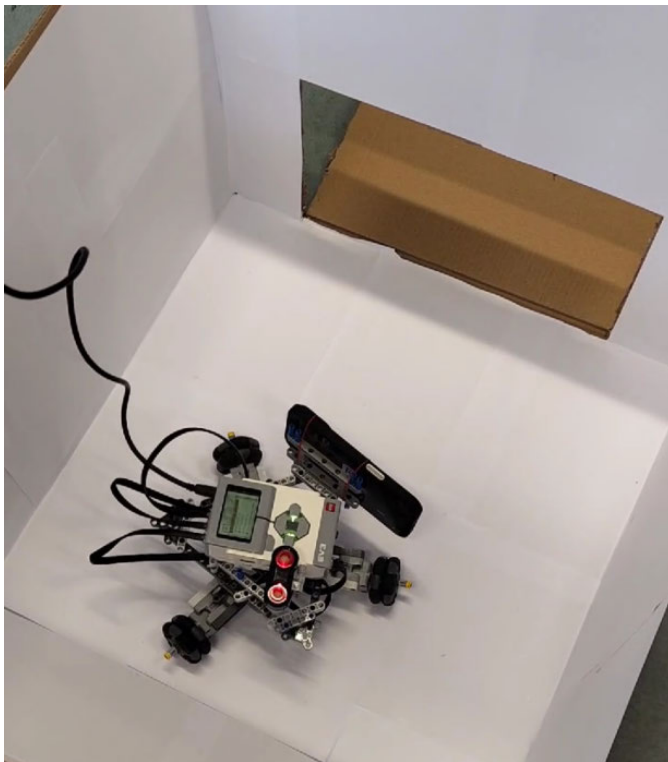


Abbildung 3. Zweite Konstruktion auf drei Rädern mit drei Motoren und Smartphone als Webcam, Fahrzeug in gebauter Testumgebung noch ohne schwarz umrandetem Ausgang

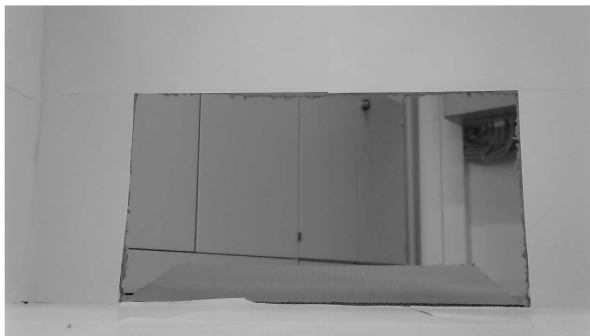


Abbildung 4. Graustufenbild des Ausgangs nach `rgb2gray()` Befehl

Decke unmittelbar über dem Fahrzeug befindet, da dies darauf hindeutet, dass die Maus in dem Moment den Raum bereits verlässt und weiter gerade aus fahren muss. Sollte dies nicht der Fall sein, führt die Maus eine kleine Rotation gegen den Uhrzeigersinn um ihre eigene Achse durch, um weiterhin im Raum nach einem Ausgang zu suchen. Sobald ein gefundenes Rechteck als potentieller Ausgang erkannt wurde, wird aus der Bounding Box der geometrische Mittelpunkt ermittelt, anhand dessen Position drei Fälle unterschieden werden. Befindet sich der Mittelpunkt auch im Zentrum des Kamerablickfeldes, so wird davon ausgegangen, dass der Ausgang frontal vor der Maus liegt und es wird eine Vorwärtsbewegung durchgeführt. Als Zentrum gewertet wird ein Mittelpunkt, dessen X-Koordinate sich höchstens 20 % links oder rechts vom



Abbildung 5. Ausgang nach Kantenerkennung mit Canny-Methode und Schwellwert von 0,7

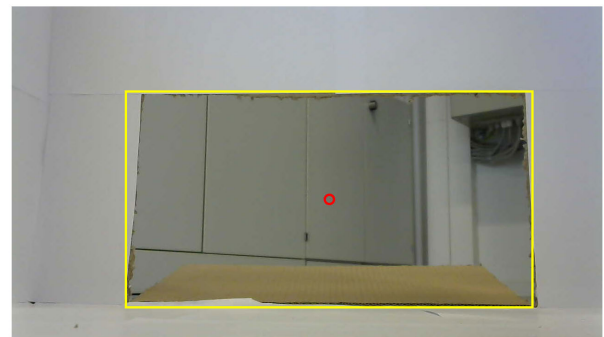


Abbildung 6. Ausgang mit Bounding Box (gelb) und geometrischer Mitte (rot)

Mittelpunkt des Frames befindet, wobei hier ein Kameraoffset an beide Bereichsgrenzen aufaddiert wird, um den negativen Einfluss der nicht vollständig mittig positionierten Kamera zu verringern. Befindet sich der Mittelpunkt des Ausgangs im rechten oder linken Bereich des Blickfelds, so werden durch kleine entgegengesetzte Drehbewegung die Maus auf diesen gefundenen Ausgang ausgerichtet. Da in jedem Durchlauf die nächste Bewegung neu berechnet wird, regelt die Maus bei Bedarf ihre Blickrichtung eigenständig auf den Ausgang, wodurch auch Ungenauigkeiten der Motoren weniger zu Gewicht fallen. Diese werden angesteuert, in dem zuerst bei allen Motoren der gewünschte Drehzahlwert (`TachoValue`) sowie deren Leistung konfiguriert, daraufhin alle Motoren gestartet werden und auf die Beendigung der Bewegung gewartet wird.

#### IV. ERGEBNISDISKUSSION

Im Allgemeinen hat sich in Tests gezeigt, dass sowohl die vorgestellte Idee der Ausgangserkennung als auch das Verlassen des Raumes mit der finalen Konstruktion funktionieren. Die Maus findet den Ausgang oft innerhalb von Sekunden und bewegt sich entsprechend auf ihn zu. Auch die eigene Ausrichtung der Maus auf den Ausgang und dessen stetige Korrektur bei Annäherung lassen sich beobachten. Die Idee das Fahrzeug kabellos zu betreiben funktionierte zwar bei der Webcam, allerdings nicht für den EV3-Baustein, da dieser Bluetooth Verbindungsprobleme aufwies. Es wird vermutet, dass die

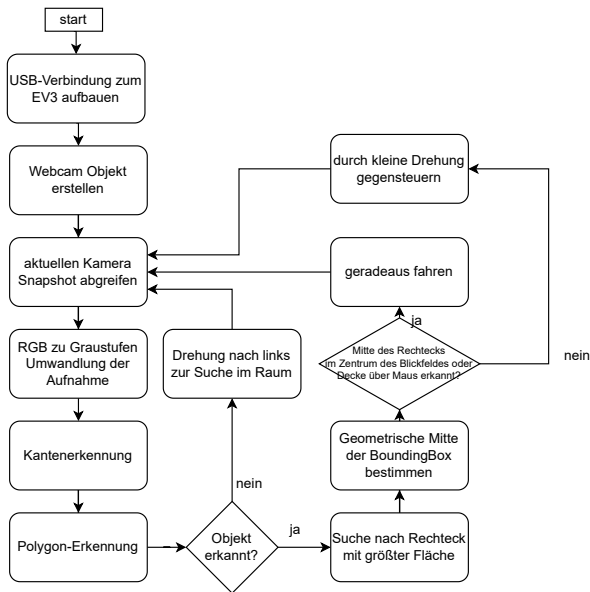


Abbildung 7. Vereinfachtes Ablaufdiagramm

höchste vom EV3 unterstützte Bluetooth-Version 2.1 [5] die Ursache für die Verbindungsprobleme mit heutiger moderner Hardware ist. Der Einsatz der Smartphone-Kamera als Webcam verbesserte die Wahrnehmen des Ausgangs signifikant und ermöglichte dies auch unter schwierigeren Lichtverhältnissen. Auch wenn der Ausgang mit hoher Zuverlässigkeit gefunden wird, sind falsch positive Erkennungen der Maus das größte Problem, da diese die Maus vom vorgesehen Weg abbringen und die Maus in diesen Fällen oft mit Seitenwänden kollidiert. Dieses Problem ist besonders präsent, wenn die Maus sich unmittelbar vor dem Ausgang befindet und das Kamerablickfeld überwiegend die Außenumgebung wahrnimmt.

Eine selbstgebaute Testumgebung mit weißen Wänden sowie einem schwarz markierten Ausgang sowie das feinjustieren der Kantenerkennungsparameter und Filterkriterien hält diese Fehlerkennungen in Grenzen, kann diese allerdings nicht vollständig verhindern. Hinzu kommt Bewegungsunschärfe der Kamera, dessen Einfluss jedoch minimiert wird, in dem das Fahrzeug nach jeder ausgeführten Bewegung für 300 Millisekunden ruht, um die Schwingungen an der Webcam ausklingen zu lassen. Es wurde beobachtet, dass die Maus häufiger beim Verlassen der Testumgebung mit den Seitenwänden kollidierte, was sich auf eine breite Fahrzeugkonstruktion der Maus, den physikalischen Kamera-Offset des Smartphones und zu groben

Bewegungen erklären lässt. Korrekturen der Bereichsgrenzen des mittleren Bereichs im Blickfeld sowie eine Verringerung der zurückgelegten Strecke bei Vorwärtsbewegungen verringerte nicht nur die Anzahl der Kollisionen mit der Wand, sondern erhöhte zusätzlich die Erkennungsrate der Decke am Ausgang durch den Ultraschallsensor.

## V. ZUSAMMENFASSUNG UND FAZIT

Das Projekt „Die Maus“ befasst sich mit der Entwicklung eines autonomen Fahrzeugs, welches mithilfe klassischer Bildverarbeitung eigenständig einen Ausgang in einem Raum erkennt und diesen verlässt. Das Fahrzeug basiert auf LEGO Mindstorms und wird von einem EV3-Baustein gesteuert, während eine Webcam Bilddaten liefert, die in MATLAB und der ComputerVisionToolbox verarbeitet werden. Eine Kombination aus Kantenerkennung und Polygonerkennung hilft dem System bei der Suche nach einem rechteckigen Ausgang und richtet sich auf diesen aus. Die erste Konstruktion mit vier Rädern erwies sich als nicht ideal, da Drehungen um die eigene Achse viel Raum benötigten und sich als zu ungenau herausstellten, weshalb eine zweite Version mit drei omnidirektionalen Rädern entwickelt wurde, die präzisere Drehbewegungen ermöglichte.

Während der Tests zeigte sich, dass das System den Ausgang zuverlässig findet, jedoch durch falsch-positive Erkennungen und Bewegungsunschärfen beeinträchtigt wurde, die durch Anpassungen minimiert werden konnten. Ein Ultraschallsensor unterstützt zudem das Verlassen des Raumes, indem erkannt wird, ob sich eine Decke über der Maus befindet. Das Projekt zeigt, dass autonome Navigation mittels klassischer Bildverarbeitung realisierbar ist, jedoch ein Bedarf nach weiteren Optimierungen in der Erkennungsgenauigkeit besteht. Als Fazit lässt sich festhalten, dass sich mit dem EV3 und einer MATLAB-Umgebung bereits mit wenigen Aktoren und Sensoren komplexere Systeme und Roboter realisieren lassen, die in der Lage sind reale Probleme zu lösen.

## LITERATURVERZEICHNIS

- [1] EVOVIU: *Klassische Bildverarbeitung vs Bildverarbeitung mit KI*. <https://www.evoviu.de/klassische-bv-vs-bv-mit-ki/>
- [2] RWTH AACHEN: *GITLAB: ev3-toolbox-matlab*. <https://git.rwth-aachen.de/mindstorms/ev3-toolbox-matlab>
- [3] MATHWORKS: *MATLAB Documentation, edge*. <https://de.mathworks.com/help/images/ref/edge.html>
- [4] ULTRALYTICS: *Bounding Box*. <https://www.ultralytics.com/de/glossary/bounding-box>
- [5] EV3DEV: *EV3 Bluetooth Module*. <https://www.ev3dev.org/docs/kernel-hackers-notebook/ev3-bluetooth/>