ROS 2 und Webots in der Lehre

ROScon DE'23 Karlsruhe – Track: Bildung

Mark O. Mints, Peer Neubert

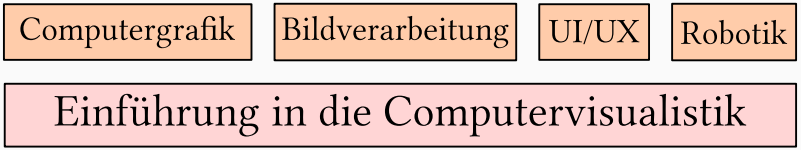
23. November 2023

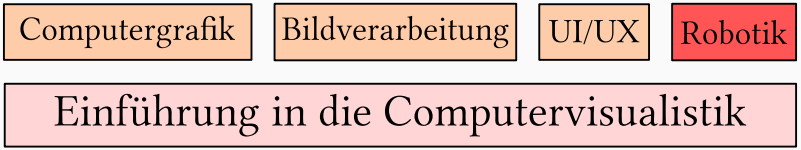
Universität Koblenz
Institut für Computervisualistik
Arbeitsgruppe Intelligente Autonome Systeme

Inhalt

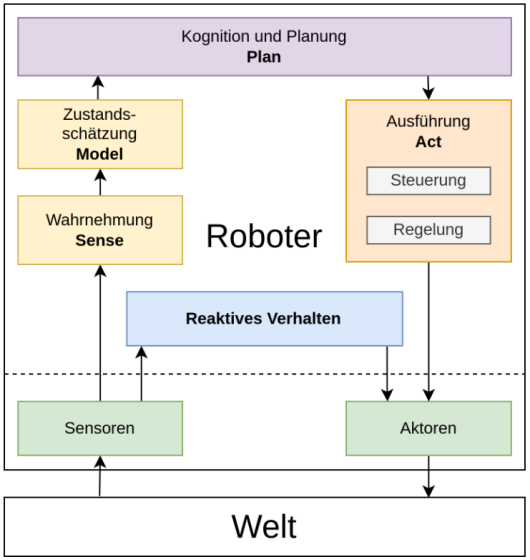
1. Lehr- und Lernziele
2. Einsatz von Webots
3. VacuSim: Software-Projekt
4. Ergebnisse der Studierenden
5. Lehrevaluation
6. Fazit

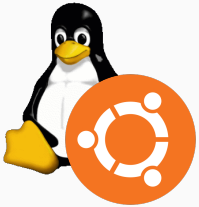
Lehr- und Lernziele


Computergrafik
Bildverarbeitung
UI/UX
Robotik
Einführung in die Computervisualistik


Computergrafik
Bildverarbeitung
UI/UX
Robotik
Einführung in die Computervisualistik

Theorie: „Big-Picture“ Robotik






HUMBLE
HAWKSBILL

Bewältigung der Lernkurve

1. Einführung

1. Einführung

1. Installation Ubuntu 22.04 (VM) und ROS 2 Humble

1. Einführung

1. Installation Ubuntu 22.04 (VM) und ROS 2 Humble
2. Erarbeitung der Konzepte von Topics und Services

Bewältigung der Lernkurve

1. Einführung

1. Installation Ubuntu 22.04 (VM) und ROS 2 Humble
2. Erarbeitung der Konzepte von Topics und Services
3. Einarbeitung in Humble-Dokumentation

Bewältigung der Lernkurve

1. Einführung

1. Installation Ubuntu 22.04 (VM) und ROS 2 Humble
2. Erarbeitung der Konzepte von Topics und Services
3. Einarbeitung in Humble-Dokumentation

2. Setup

Bewältigung der Lernkurve

1. Einführung

1. Installation Ubuntu 22.04 (VM) und ROS 2 Humble
2. Erarbeitung der Konzepte von Topics und Services
3. Einarbeitung in Humble-Dokumentation

2. Setup

1. Installation von Webots

Bewältigung der Lernkurve

1. Einführung

1. Installation Ubuntu 22.04 (VM) und ROS 2 Humble
2. Erarbeitung der Konzepte von Topics und Services
3. Einarbeitung in Humble-Dokumentation

2. Setup

1. Installation von Webots
2. Installation des Roboter-Driver ROS-Pakets

Bewältigung der Lernkurve

1. Einführung

1. Installation Ubuntu 22.04 (VM) und ROS 2 Humble
2. Erarbeitung der Konzepte von Topics und Services
3. Einarbeitung in Humble-Dokumentation

2. Setup

1. Installation von Webots
2. Installation des Roboter-Driver ROS-Pakets
3. Erstellen eines eigenen ROS-Pakets und Programmierung der ersten Node

Bewältigung der Lernkurve

1. Einführung

1. Installation Ubuntu 22.04 (VM) und ROS 2 Humble
2. Erarbeitung der Konzepte von Topics und Services
3. Einarbeitung in Humble-Dokumentation

2. Setup

1. Installation von Webots
2. Installation des Roboter-Driver ROS-Pakets
3. Erstellen eines eigenen ROS-Pakets und Programmierung der ersten Node

3. „Staubsauger-Wettbewerb“

Implementieren Sie eine Robotersteuerung, sodass ein Staubsaugerroboter in einer simulierten Testumgebung möglichst effizient den Boden reinigt.

Einsatz von Webots

Warum eigentlich Webots?

I. Einsatz auf unbekannter Hardware

I. Einsatz auf unbekannter Hardware

II. Einsatz in virtueller Maschine

Vergleich von Simulationsumgebungen

Weighted metrics value evaluation results.

Metric name	Weight	CoppeliaSim	Gazebo	MORSE	Webots
Free to use	4	1.000	1.000	1.000	1.000
Open source	2	0	1.000	1.000	1.000
ROS integration	6	0.800	1.000	1.000	0.800
Programming languages	3	1.000	0.667	0.333	1.000
UI functionality	6	1.000	1.000	0.333	1.000
Model format support	4	1.000	1.000	0	0.250
Physics engine support	3	1.000	1.000	0	0
Real time factor	4	0.914	1.000	0.789	0.849
Average load CPU efficiency	2	0.364	0.174	0.333	1.000
Intense load CPU efficiency	2	0.583	0.304	0.583	1.000
IMU angular velocity accuracy	10	0.875	1.000	0.792	0.867
IMU linear acceleration accuracy	10	1.000	0.713	0.424	0.736
Total	**56**	**49.100**	**49.087**	**32.148**	**44.226**

Farley, A. et al. (2022). How to pick a mobile robot simulator [FWM22]

Vergleich von Simulationsumgebungen

Weighted metrics value evaluation results.

Metric name	Weight	CoppeliaSim	Gazebo	MORSE	Webots
Free to use	4	1.000	1.000	1.000	1.000
Open source	2	0	1.000	1.000	1.000
ROS integration	6	0.800	1.000	1.000	0.800
Programming languages	3	1.000	0.667	0.333	1.000
UI functionality	6	1.000	1.000	0.333	1.000
Model format support	4	1.000	1.000	0	0.250
Physics engine support	3	1.000	1.000	0	0
Real time factor	4	0.914	1.000	0.789	0.849
Average load CPU efficiency	2	0.364	0.174	0.333	1.000
Intense load CPU efficiency	2	0.583	0.304	0.583	1.000
IMU angular velocity accuracy	10	0.875	1.000	0.792	0.867
IMU linear acceleration accuracy	10	1.000	0.713	0.424	0.736
Total	**56**	**49.100**	**49.087**	**32.148**	**44.226**

Farley, A. et al. (2022). How to pick a mobile robot simulator [FWM22]

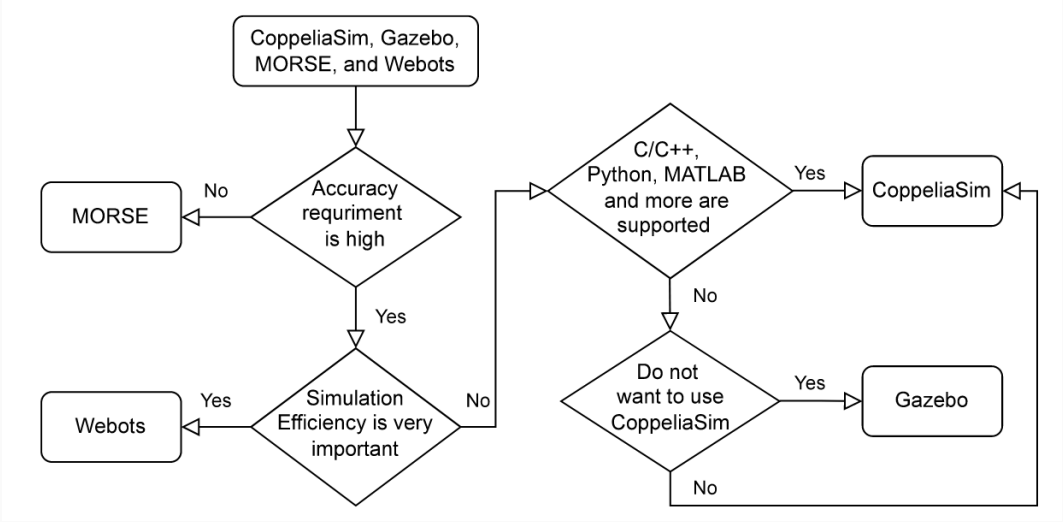
Vergleich von Simulationsumgebungen

Summary of the quantitative metrics evaluation results.

Metric name	CoppeliaSim	Gazebo	MORSE	Webots
Real time factor	0.973	1.064	0.839	0.903
Average load CPU efficiency	11%	23%	12%	4%
Intense load CPU efficiency	12%	23%	12%	7%

Farley, A. et al. (2022). How to pick a mobile robot simulator [FWM22]

Vergleich von Simulationsumgebungen



Farley, A. et al. (2022). How to pick a mobile robot simulator [FWM22]

Einsatz von Webots

Gründe für die Wahl von Webots:

Gründe für die Wahl von Webots:

1. Ressourcenschonend

Gründe für die Wahl von Webots:

1. Ressourcenschonend
2. Rendering-Probleme von Gazebo in virtuellen Maschinen

Gründe für die Wahl von Webots:

1. Ressourcenschonend
2. Rendering-Probleme von Gazebo in virtuellen Maschinen
3. Aufgeräumte UI mit ingeriertem Texteditor

Einsatz von Webots

Gründe für die Wahl von Webots:

1. Ressourcenschonend
2. Rendering-Probleme von Gazebo in virtuellen Maschinen
3. Aufgeräumte UI mit ingeriertem Texteditor
4. Bietet einen zu Gazebo ähnlich großen Funktionsumfang

Einsatz von Webots

Gründe für die Wahl von Webots:

1. Ressourcenschonend
2. Rendering-Probleme von Gazebo in virtuellen Maschinen
3. Aufgeräumte UI mit ingeriertem Texteditor
4. Bietet einen zu Gazebo ähnlich großen Funktionsumfang
5. Persönliche Neugier

VacuSim: Software-Projekt

VacuSim: Paketübersicht

VacuSim: Paketübersicht

universität koblenz
weiterdenken

`vacusim_robot_driver`

VacuSim: Paketübersicht

universität koblenz
weiterdenken

vacusim_robot_driver

1. PROTO-Beschreibung eines angepasten *iRobot Create*

VacuSim: Paketübersicht

universität koblenz
weiterdenken

vacusim_robot_driver

1. `PROTO`-Beschreibung eines angepasten *iRobot Create*
2. `Driver-Node (Python):` Schnittstelle zwischen ROS 2 und Webots

VacuSim: Paketübersicht

universität koblenz
weiterdenken

`vacusim_robot_driver`

1. `PROTO`-Beschreibung eines angepasten *iRobot Create*
2. `Driver-Node (Python):` Schnittstelle zwischen ROS 2 und Webots
3. `PROTO`-Beschreibungen von verschiedenen *Welten*

VacuSim: Paketübersicht

`vacusim_robot_driver`

1. `PROTO`-Beschreibung eines angepasten *iRobot Create*
2. `Driver-Node (Python):` Schnittstelle zwischen ROS 2 und Webots
3. `PROTO`-Beschreibungen von verschiedenen *Welten*
4. `Supervisor Controller (C):` *Allwissender* Webots Controller zur Evaluation

VacuSim: Paketübersicht

universität koblenz
weiterdenken

vacusim_robot_driver

1. PROTO-Beschreibung eines angepasten *iRobot Create*
2. Driver-Node (Python): Schnittstelle zwischen ROS 2 und Webots
3. PROTO-Beschreibungen von verschiedenen *Welten*
4. Supervisor Controller (C): *Allwissender* Webots Controller zur Evaluation

vacusim_robot_interfaces

VacuSim: Paketübersicht

universität koblenz
weiterdenken

vacusim_robot_driver

1. `PROTO`-Beschreibung eines angepasten *iRobot Create*
2. `Driver-Node (Python):` Schnittstelle zwischen ROS 2 und Webots
3. `PROTO`-Beschreibungen von verschiedenen *Welten*
4. `Supervisor Controller (C):` *Allwissender* Webots Controller zur Evaluation

vacusim_robot_interfaces

1. Sensor-Messages: `Bumper.msg, Distance.msg`

VacuSim: Paketübersicht

universität koblenz
weiterdenken

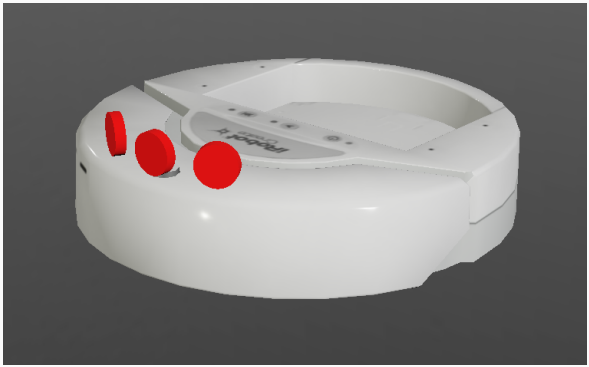
vacusim_robot_driver

1. `PROTO`-Beschreibung eines angepasten *iRobot Create*
2. `Driver-Node (Python):` Schnittstelle zwischen ROS 2 und Webots
3. `PROTO`-Beschreibungen von verschiedenen *Welten*
4. `Supervisor Controller (C):` *Allwissender* Webots Controller zur Evaluation

vacusim_robot_interfaces

1. Sensor-Messages: `Bumper.msg, Distance.msg`
2. Bewengungs-Services: `Drive.srv, Turn.srv`

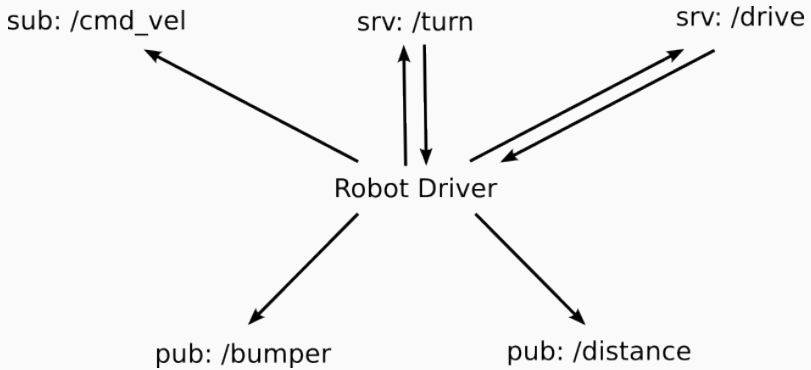
VacuSim: Custom iRobot Create



Angepassert *iRobot Create*[1] mit zusätzlichen Distanzsensoren (45° Öffnungswinkel)

[1] https://github.com/cyberbotics/webots/blob/released/projects/robots/irobot/create/protos/Create.proto

VacuSim: Robot Driver



VacuSim: Robot Driver – Fahrmodi

Kontinuierlich über Geschwindigkeit (`/cmd_vel`)

Der Roboter wird über lineare und Winkelgeschwindigkeit durch die Arena gesteuert.

VacuSim: Robot Driver – Fahrmodi

Kontinuierlich über Geschwindigkeit (`/cmd_vel`)

Der Roboter wird über lineare und Winkelgeschwindigkeit durch die Arena gesteuert.

Diskret über Distanz (`/turn` und `/drive`)

Der Roboter bewegt sich immer nur so lange, bis die gewünschte Distanz erreicht ist bzw. der gewünschte Winkel erreicht wurde.

VacuSim: Robot Driver – Sensoren

Bumper (`/bumper`)

Gibt für linken und rechten Tastsensor jeweils einen `bool`-Wert zurück.

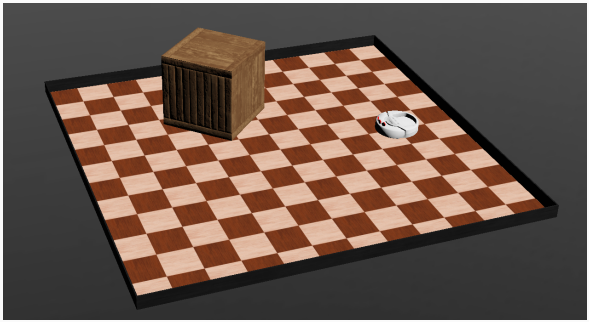
Bumper (`/bumper`)

Gibt für linken und rechten Tastsensor jeweils einen `bool`-Wert zurück.

Distance (`/distance`)

Gibt für linken, mittleren und rechten Distanzsensor jeweils einen `float`-Wert zurück.

VacuSim-Welten: Simple Arena



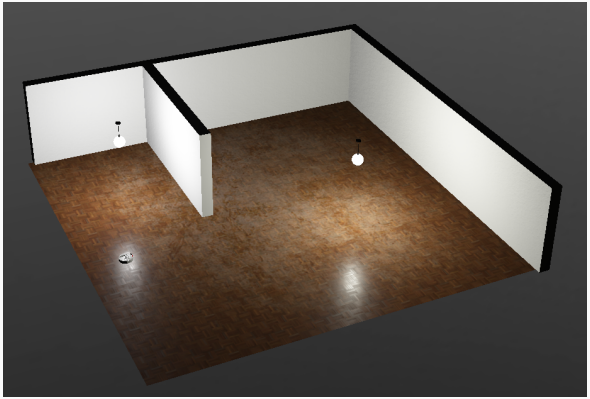
Einfache Szene zum Experimentieren und Testen.

VacuSim-Welten: Apartment

Angepasste *iRobot Create* Beispielwelt `create.wbt` [2]

[2] Nach Webots-Installtion zu finden unter: `$WEBOTS_HOME/projects/robots/irobot/create/worlds`

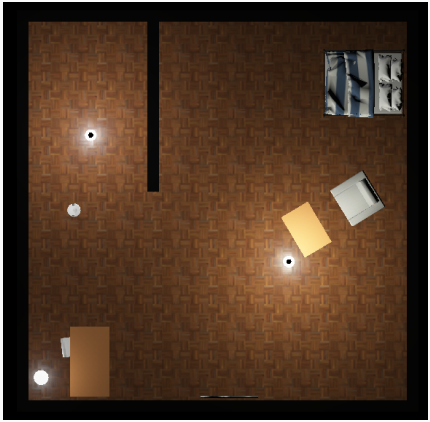
VacuSim-Welten: Leeres Apartment

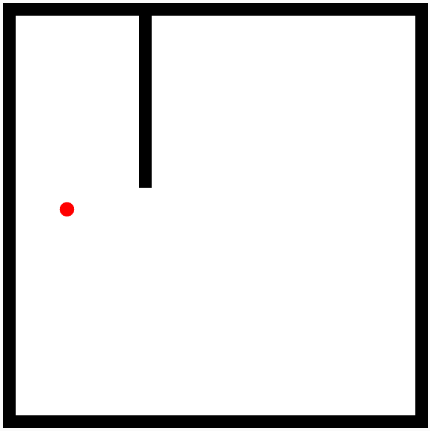


Angepasste *iRobot Create* Beispielwelt `create.wbt` [3]

[3] Nach Webots-Installtion zu finden unter: `$WEBOTS_HOME/projects/robots/irobot/create/worlds`

VacuSim-Welten: Apartment-Karte





VacuSim: Supervisor-Controller

Rendering

Rendering

Der Controller basiert auf Code aus der Beispielwelt `create.wbt`:

Rendering

Der Controller basiert auf Code aus der Beispielwelt `create.wbt`:

Visuelles Feedback beim Ausführen der Simulation.

Rendering

Der Controller basiert auf Code aus der Beispielwelt `create.wbt`:

Visuelles Feedback beim Ausführen der Simulation.

Evaluation

Rendering

Der Controller basiert auf Code aus der Beispielwelt `create.wbt`:

Visuelles Feedback beim Ausführen der Simulation.

Evaluation

- 600 Sekunden Laufzeit

Rendering

Der Controller basiert auf Code aus der Beispielwelt `create.wbt`:

Visuelles Feedback beim Ausführen der Simulation.

Evaluation

- 600 Sekunden Laufzeit
- Berechnung der abgefahrenen/„gereinigten“ Fläche

Rendering

Der Controller basiert auf Code aus der Beispielwelt `create.wbt`:

Visuelles Feedback beim Ausführen der Simulation.

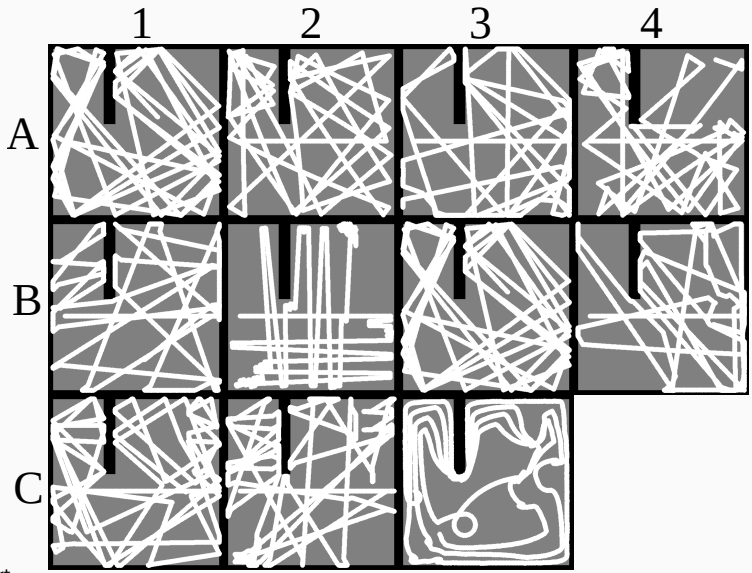
Evaluation

- 600 Sekunden Laufzeit
- Berechnung der abgefahrenen/„gereinigten“ Fläche
- Ausgabe:
  - Tabelle: Reinigungsfortschritt alle n Sekunden
  - Karte des abgefahreren Bereichs

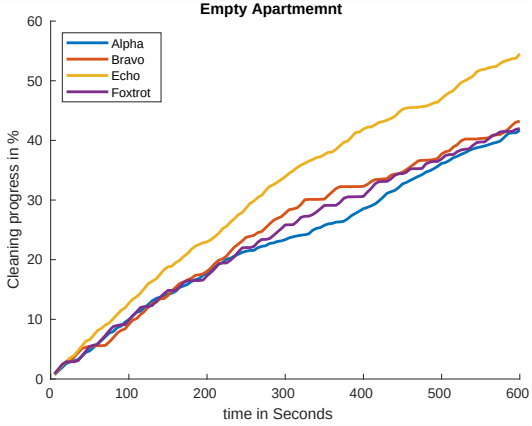
VacuSim: Supervisor Controller – Rendering

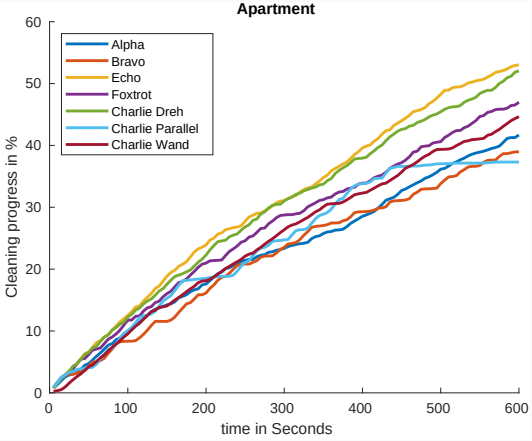
Ergebnisse der Studierenden

Ergebniskarten



Reinigungsfortschritt





Lehrevaluation

Lehrevaluation – Eigene Einschätzung

Lehrevaluation – Eigene Einschätzung

Positiv

- Thema Robotik mehr ins Licht gerückt

Lehrevaluation – Eigene Einschätzung

Positiv

- Thema Robotik mehr ins Licht gerückt
- Studierende haben kreative Lösungen vorgestellt

Lehrevaluation – Eigene Einschätzung

Positiv

- Thema Robotik mehr ins Licht gerückt
- Studierende haben kreative Lösungen vorgestellt
- Persönlicher Einstieg in ROS 2

Lehrevaluation – Eigene Einschätzung

Positiv

- Thema Robotik mehr ins Licht gerückt
- Studierende haben kreative Lösungen vorgestellt
- Persönlicher Einstieg in ROS 2

Negativ

Lehrevaluation – Eigene Einschätzung

Positiv

- Thema Robotik mehr ins Licht gerückt
- Studierende haben kreative Lösungen vorgestellt
- Persönlicher Einstieg in ROS 2

Negativ

- Initial sehr viel Aufwand, einen niedrigschwelligen Einstieg zu gewährleisten

Lehrevaluation – Eigene Einschätzung

Positiv

- Thema Robotik mehr ins Licht gerückt
- Studierende haben kreative Lösungen vorgestellt
- Persönlicher Einstieg in ROS 2

Negativ

- Initial sehr viel Aufwand, einen niedrigschwelligen Einstieg zu gewährleisten
- Individuelle Hilfestellung oft benötigt

Lehrevaluation – Feedback der Studierenden

Lehrevaluation – Feedback der Studierenden

Positiv

- Interesse am Thema Robotik gefunden

Lehrevaluation – Feedback der Studierenden

universität koblenz
weiterdenken

Positiv

- Interesse am Thema Robotik gefunden
- Aufgaben-Szenario hat zur Programmierung motiviert

Lehrevaluation – Feedback der Studierenden

Positiv

- Interesse am Thema Robotik gefunden
- Aufgaben-Szenario hat zur Programmierung motiviert
- Spannendster Kurs in diesem Semester

Lehrevaluation – Feedback der Studierenden

Positiv

- Interesse am Thema Robotik gefunden
- Aufgaben-Szenario hat zur Programmierung motiviert
- Spannendster Kurs in diesem Semester

Negativ

Lehrevaluation – Feedback der Studierenden

Positiv

- Interesse am Thema Robotik gefunden
- Aufgaben-Szenario hat zur Programmierung motiviert
- Spannendster Kurs in diesem Semester

Negativ

- Sehr schwerer Einstieg:
  - Zum ersten mal ein OS installiert
  - Zum ersten mal Linux genutzt
  - Zum ersten mal Python programmiert

Lehrevaluation – Feedback der Studierenden

Positiv

- Interesse am Thema Robotik gefunden
- Aufgaben-Szenario hat zur Programmierung motiviert
- Spannendster Kurs in diesem Semester

Negativ

- Sehr schwerer Einstieg:
  - Zum ersten mal ein OS installiert
  - Zum ersten mal Linux genutzt
  - Zum ersten mal Python programmiert
- Kein Nutzen von ROS 2 im weiteren Studienverlauf

Fazit

Fazit

Fazit

1. Steilere Lernkurve als erwartet

Fazit

1. Steilere Lernkurve als erwartet
2. Linux umgehen?
   - Native Installation von ROS 2 und Webot auf Windows/MacOS?
   - RoboStack, Fischer et al. [FVT+21]
   - Cloudlösung?

Fazit

1. Steilere Lernkurve als erwartet
2. Linux umgehen?
   - Native Installation von ROS 2 und Webot auf Windows/MacOS?
   - RoboStack, Fischer et al. [FVT+21]
   - Cloudlösung?
3. Erweiterung der Benchmark-Kriterien

Projekt Repository[a]

[a] https://gitlab.uni-koblenz.de/intas/vacusim

Fazit

1. Steilere Lernkurve als erwartet
2. Linux umgehen?
   - Native Installation von ROS 2 und Webot auf Windows/MacOS?
   - RoboStack, Fischer et al. [FVT+21]
   - Cloudlösung?
3. Erweiterung der Benchmark-Kriterien

Projekt Repository[a]

[a] https://gitlab.uni-koblenz.de/intas/vacusim

Vielen Dank!

Quellen

Quellen i

Fischer, Tobias ; Vollprecht, Wolf ; Traversaro, Silvio ; Yen, Sean ; Herrero, Carlos ; Milford, Michael:
A RoboStack Tutorial: Using the Robot Operating System Alongside the Conda and Jupyter Data Science Ecosystems.
In: *IEEE Robotics and Automation Magazine* (2021).
http://dx.doi.org/10.1109/MRA.2021.3128367. –
DOI 10.1109/MRA.2021.3128367

Farley, Andrew ; Wang, Jie ; Marshall, Joshua A.:
How to pick a mobile robot simulator: A quantitative comparison of CoppeliaSim, Gazebo, MORSE and Webots with a focus on accuracy of motion.
In: *Simulation Modelling Practice and Theory* 120 (2022), 102629.
http://dx.doi.org/https://doi.org/10.1016/j.simpat.2022.102629. –
DOI https://doi.org/10.1016/j.simpat.2022.102629