Project Name	Turtlebot Fleet Management
Online team meeting	https://fau.zoom.us/j/65679458667
Production system (if any)	...
Test system (if any)	...
GitHub repository	https://github.com/amosproj/amos2022ss03-turtlebot-fleet-management
GitHub kanban board (project)	https://github.com/amosproj/amos2022ss03-turtlebot-fleet-management/projects/1
Team T-shirt (black)	https://www.shirtinator.de/loadBasket/Gip4U1-D_O7
Sprint Release Guide	https://docs.google.com/document/d/1rddiixyKOZ-jsgIK0DWDUSfKee97NrXL5HkiXUdLapk/edit#heading=h.3qy4m6dkebbo

Last Name	First Name	GitHub User Name	Email Address
Vogler	Tim	cat24max	tim.vogler@fau.de
Scherbel	Sebastian	Sebastian2023	sebastian.scherbel@fau.de
Petersen	Jonas	JonasPetersenFAU	jonas.petersen@fau.de
Blöcher	Meike	MeikeBloecher	meike.bloecher@fau.de
Markert	Niklas	nmarkert	niklas.markert@fau.de
Ramaiya	Umang Bharatkumar	UmangBR	umang.ramaiya@fau.de
Janjua	Muhammad Usman	usmanjanjua786	usman.janjua@fau.de
Moorthy	Venkatesh Kumar	Venkatesh770	venkatesh.kumar.moorthy@fau.de
Alekseenko	Ekaterina	ekaterinaaleksee	ekaterina.alekseenko@fau.de

Goals	Achieve goal of industry partner
	Foster and atmosphere of learning
	Everybody has to have fun during the course
Meeting norms	Eveybody shows up on-time (Wednesday 12:30 pm)
	Meeting with business partners is once a week
	We do not interrupt each other
	There shall be a friendly atmosphere
	We are fair to other team members (pair programming,...)
	Every idea is welcome
Working norms	Everyone contributes regularly
	We take criticism positively and try to learn from it
	We value quality over quantity
Coordination norms	Every job has a responsible person
	We volunteer for jobs
	The responsible person has to be marked in the feature board
	Job assignment: First come, first serve!
Communication norms	We follow the Chatham house rules
	We use Slack for formal infos to the team & Whatsapp for informal information
	We check Slack at least once a day
Consideration norms	We discuss disagreement openly
	We vote for a final resolution
	Everyone has the same voting rights
Cont. improvement norms	We jointly review the happiness index
	You must raise insufficient quality issues
	Everybody has to send a stand-up mail at least twice a week
	We fill out the happiness index at the end of the meeting
	A continous improvement has to be visual
Rewards	We celebrate a succesful release
	After a successful sprint release with use clapping reaction on Zoom
Sanctions	You must raise clear violations of the team contract
	Consequences for violations of the team contract are discussed by the team

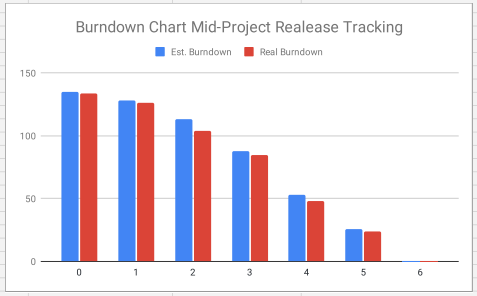
#	Meeting Day	Uni	Comment	Product Owner	Software Developer	Release Manager	Scrum Master
1	2022-04-27			Umang Ramaiya/ Jonas Petersen	Everyone else	N/A	Ekaterina Alekseenko
2	2022-05-04			Umang Ramaiya/ Jonas Petersen	Everyone else	N/A	Ekaterina Alekseenko
3	2022-05-11	Yes		Umang Ramaiya/ Jonas Petersen	Everyone else	Tim Vogler	Ekaterina Alekseenko
4	2022-05-18			Umang Ramaiya/ Jonas Petersen	Everyone else	Meike Blöcher	Ekaterina Alekseenko
5	2022-05-25	Yes		Umang Ramaiya/ Jonas Petersen	Everyone else	Sebastian Scherbel	Ekaterina Alekseenko
6	2022-06-01			Umang Ramaiya/ Jonas Petersen	Everyone else	Niklas Markert	Ekaterina Alekseenko
7	2022-06-08	Yes	Mid-term due	Umang Ramaiya/ Jonas Petersen	Everyone else	Venkatesh Kumar	Ekaterina Alekseenko
8	2022-06-15			Umang Ramaiya/ Jonas Petersen	Everyone else	Muhammad Usman Janjua	Ekaterina Alekseenko
9	2022-06-22			Umang Ramaiya/ Jonas Petersen	Everyone else	Tim Vogler	Ekaterina Alekseenko
10	2022-06-29	Yes		Umang Ramaiya/ Jonas Petersen	Everyone else	Meike Blöcher	Ekaterina Alekseenko
11	2022-07-06			Umang Ramaiya/ Jonas Petersen	Everyone else	Sebastian Scherbel	Ekaterina Alekseenko
12	2022-07-13			Umang Ramaiya/ Jonas Petersen	Everyone else	Niklas Markert	Ekaterina Alekseenko
13	2022-07-20	Yes		Umang Ramaiya/ Jonas Petersen	Everyone else	Venkatesh Kumar	Ekaterina Alekseenko
14	2022-07-27		Demo day!	Umang Ramaiya/ Jonas Petersen	Everyone else	Muhammad Usman Janjua	Ekaterina Alekseenko
15	2022-08-03		Retrospective	Umang Ramaiya/ Jonas Petersen	Everyone else	N/A	Ekaterina Alekseenko

Product Vision	Project Mission
The vision is to have a management system which helps fulfill daily tasks in a smart and intelligent way. TurtleBots are automated guided vehicles (AGVs) which assist humans without their intervention. The TurtleBot fleet management system is envisioned to bring intelligence to a fleet of these AGVs. It manages every robot to increase efficiency effectively.	The mission is to develop three key components namely, a fleet management system, an on-robot navigation system and a user interface along with interfaces to have an intra-component communication. The fleet management system has to manage TurtleBots (AGVs) on a defined circular course. The TurtleBots need to communicate with the fleet management using MQTT & VDA5050 and should navigate in the available physical space to deliver small goods from a home station to a particular station on a pre-planned route and reorient themselves when going off-course. An interactive user interface should provide status information for every robot.

Term	Definition
AGV / TurtleBot	Automated Guided Vehicle, a mobile robot (the TurtleBot) used in industrial applications to move materials from point A to B
AGV module for FMS	Provides the structure for an AGV and handles the correct execution of the to the AGV assigned orders.
Base	The „locked in“ route. All nodes in base are reserved and the robot is released to drive on them.
Bootstrap	A free & open-source CSS framework directed at responsive, mobile-first front-end web development. Used to have a better looking UI
Calculation function (Robot worker module)	Calculates the direction and speed for the robot to reach the destination.
Chart js	An open-source JavaScript library for data visualization, which supports eight chart types
Critical path	The path chosen by a TurtleBot to traverse from source to destination
Driver segment (Robot worket module)	Drives the robot along the line and checks whether the robot is still on the line to react accordingly.
Docker	A software platform that allows you to build, test, and deploy applications quickly
Flask	A micro web framework written in Python which is used to build the webserver
FMS	Fleet management system has the functionalities to coordinate a fleet of robots
Graph Module for FMS	Defines a storage format and provides functions for traversal
Horizon	The future route beyond base which is not yet locked or released to the AGV.
JavaScript	The programming language of the Web.
MQTT	A lightweight, publish-subscribe network protocol that transports messages between devices
MQTT Module for FMS	Establishes the MQTT connection with the broker, receives and handles incoming VDA5050 packets and sends outgoing ones
MQTT Module for TurtleBot	Establishes a MQTT connection with the broker and de/encode messages using vda5050_msgs.
OpenWRT	An open-source project for embedded operating systems. It is used for networking between TurtleBot and FMS
Paho MQTT	Eclipse Paho Python is a Python language client library which is used to build the MQTT connection
Order module for FMS	Provides the structure for a driving order and does the main part of avoiding collisions between different orders.
Publisher (Robot worker module)	Publishes the current location, speed, direction and battery status
Python3	Latest version of Python, a high-level, interpreted, general-purpose programming language
RasPi	Raspberry Pi: A credit-card sized computer who's OS acts as a powerful combination to create smart robots
ROS	Robot Operating System, a framework that helps researchers and developers build and reuse code between robotics applications
Sick LiDAR LOC	A software for determining the position of automated guided vehicles (AGVs)
SMET	Sick Map Engineering Tool: helps create maps for localization
Subscriber (Robot worker module)	Gets relevant topics from the mqtt node on the robot, the localisation topic and the line measurement topic
VDA5050	A standardized interface for AGV communication
VDA5050 Module for FMS	To create VDA5050 JSON strings and read VDA5050 packets
VMap Ingress Module for FMS	Reads scanned LIDAR map file and converts it to FMS Python graph
Vue.js	An open-source model–view–viewmodel front end JavaScript framework for building user interfaces and single-page applications.
Webserver Module for FMS	To recieve and execute requests from User Interface
Worker Module for FMS	Distributes pending orders to the AGVs and provides functions to get the needed informations for the UI.
Worker Module for Robot	The worker module fulfills incoming orders and sends position and status updates back to the FMS. It has a subscriber & publisher, a calculation function and a driver segment

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down
0			**Total**		**135**		**134**	**134**
1	**Familiarize with Project**	**Getting to know team and industry partner, project organization**			**7**		**8**	**126**
			Meeting with industry partner	1		1		
			Additional team meeting of SDs	1		1		
			Choice of programming language	1		1		
			Get material from industry partner	1		1		
			Designing team logo and T-shirts	2		2		
			Find room to work	1		2		
2	**Initial setup**	**Getting to know further software requirements from industry partner & setting up initial software**			**15**		**22**	**104**
			Software Architecture	2		2		
			Bill of materials	1		1		
			Get familiar with software/turtlebot	2		3		
			Get used to sensors and algorithm	2		3		
			Get familiar with the fleet management system	2		3		
			Get familiar with the user interface	2		3		
			Build standalone UI	2		5		
			Service code running on RasPi	2		2		
3	**First setup of TurtleBots**	**Creating a navigation course for TurtleBot**			**25**		**19**	**85**
			Setting up SMET	2		1		
			Connect TurtleBot network to the internet	2		3		
			Creating a room map	3		2		
			Evaluate and specify TurtleBot - FMS interface	3		3		
			Create a FMS Backend to Frontend interface	3		2		
			Configure virtual line sensor in lidar loc software	3		2		
			Use ROS drive to recieve line measurements	5		2		
			Brainstorm on TurtleBot modules	2		2		
			Brainstorming on FMS Modules	2		2		
			Drive the robot with joystick	5		5		
4	**Developing FMS Modules**	**Getting Modules for FMS working, clear understanding of product vision & project mission**			**35**		**37**	**48**
			[TB] Brainstorm on worker modules	3		2		
			[FMS] VDA5050 Module	3		5		
			[FMS] Graph Module	8		5		
			[FMS] VMap Ingress Module	5		5		
			Upgrade ROS to latest version	3		5		
			Drive the robot with joystick	5		5		
			FMS-MQTT Connection	1		1		
			Product Vision & Project Mission	1		1		
			Definition of Done	1		1		
			Setup commit linter and checker	2		2		
			Create MQTT Bridge for Connection Module on the TurtleBot	3		5		
5	**Development of TurtleBot worker modules and further developments for FMS modules**	**Development of FMS and TurtleBot modules, first connection tests**			**27**		**24**	**24**
			[TB] Develop the subscriber and publisher (for Worker module)	5		3		
			[TB] Develop the calculation function (for Worker module)	3		2		
			[FMS] Graph route finding algorithm	8		8		
			[FMS/TB] tests	5		5		
			Create build process video	3		3		
			Create mid-project release plan	3		3		
6	**Mid-term release and testing**	**Have a fully connected basic system, create videos and documentation**			**26**		**24**	**0**
			[TB] Develop the driver segment (for Worker module)	5		8		
			[FMS] VDA5050 packet receiver	3		3		
			Shortest path vizualization	1		2		
			Get basic system up and running	5		3		
			Initialize User/design/build documentation	3		2		
			Create video for industry partner	5		3		
			Clean up mid-project release plan	1		1		
			Create project release plan	3		2		

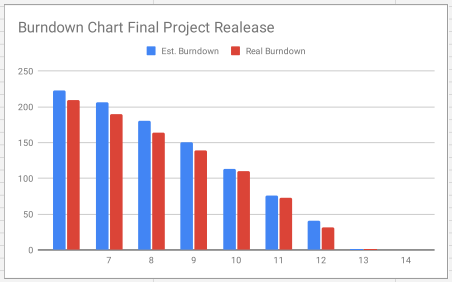
	Sprint	Sprint Theme	Est. Size	Est. Burndown	Real Size	Real Burndown
	0			**135**		**134**
	1	**Familiarize with Project**	**7**	**128**	**8**	**126**
	2	**Initial setup**	**15**	**113**	**22**	**104**
	3	**First setup of TurtleBots**	**25**	**88**	**19**	**85**
	4	**Developing FMS Modules**	**35**	**53**	**37**	**48**
	5	**Development of TurtleBot worker modules and further developments for FMS modules**	**27**	**26**	**24**	**24**
	6	**Mid-term release and testing**	**26**	**0**	**24**	**0**



#	Feature Definition of Done	Sprint Release Definition of Done	Project Release Definition of Done
	Linter & Checker were performed and passed	Everything of the sprint is merged into release candidate	User manual is written and passed review
	Code Review has been completed	Code builds without errors and tests successfull	Software documentation is written and passed review
	Code was merged in sprint release candidate	Acceptance of Product Owner for sprint release	Release candidate fulfills everything customer wants
	Updates are written in issue comments	Sprint release candidate is tagged as realse candidate	Code builds without errors and tests successfully
	Feature builds and tests successfully	Sprint release is tagged	Acceptance of Product Owner for release
	Acceptance by product owner		Code in GitHub is documented and enough information is provided
	User Story & Acceptance Criteria fulfilled		

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down
			Total		**223**			**209**
7	**Improvements & Leftovers from Releas**	**Development of worker module for FMS and further**	**enhancements of worker module (driver segment) for turtlebot**		**17**		**19**	**190**
			[TB] Debug latency issues	3		2		
			Display a map of the robots current location in graph	3		3		
			Create a new map	3		3		
			[TB] Further develop the driver segment (for Worker module)	8		8		
8	**UI: Displaying robot status info**	**Apart from robot location, it's battery status and velocity**	**is also displayed on the UI**		**26**		**26**	**164**
			[TB] Create launch file	3		3		
			[TB] Make TB driving smoother and faster	5		5		
			Displaying status info on UI	5		5		
			[TB] Push vmap to turtlebots and set position automatically	3		5		
			Improve global UI	5		3		
			[FMS] Order management	5		5		
9	**Finish basic set-up (UI, FMS, TBs) & Collision avoidance**	**Robot takes destination from UI and takes shortest path to destination on course**			**30**		**25**	**139**
			[FMS] Auto-refresh AGV status table	1		1		
			[FMS] Path Highlighting	5		5		
			[TB] Use curve feature from sick to get smoother curves	3		3		
			[FMS] Push vmap to turtlebots on startup	3		3		
			[FMS] Implement collision avoidance for multiple robots	8		5		
			[TB] Implementing the order update functionality	5		5		
			Receive order status updates from TB	5		3		
10	**Multiple turtlebot management**	**Priority/collision avoidance/ battery aspects are managed for both robots simultaneously**			**37**		**29**	**110**
			Create lessons learned list	3		2		
			[TB] Improve driving further	5		3		
			[TB] Create video of driving robot	3		2		
			[FMS] Update AGV Info Table	3		3		
			[FMS] Order safety check	5		3		
			[FMS] Active order critical path calculation function	3		3		
			[FMS] Calculate near future nodes to be locked	5		5		
			[FMS] Create new order format	5		5		
			[TB] Order abortion	5		3		
11	**Tests/Add features to new graph**	**UI has all features like old UI/Test features and debug**			**37**		**37**	**73**
			[FMS] Tests with the TBs	13		13		
			[TB] Tests with FMS	13		13		
			[TB] Automatic recharging of robots	3		3		
			[FMS] Order delegator thread	5		5		
			[FMS] Create driving simulation	3		3		
12	**Scheduling robots for recharging**	**Robots automatically recharge when battery is low +**	**testing**		**35**		**41**	**32**
			Reinstall Raspberry Pi	3		2		
			[FMS] Order abortion	3		3		
			[FMS] Automatic recharging of robots	5		5		
			[FMS/TB] Testing and stabilizing everything when working together	8		13		
			Demostrate path splitting and both robots trying to take the same path	5		8		
			Create demo day video	8		8		
			Create Demo day slide	3		2		
13	**Finalize project**	**Final touches and setup for the final project release**			**41**		**32**	**0**
			Finalize user, (technical) design, and build/deploy documentation	3		3		
			Finish final release plan	2		2		
			Create PPT for demo day	2		1		
			Demo day video setup and created new videos if required	5		2		
			[FMS] Cleanup codebase and fix any remaining bugs	8		5		
			[TB] Cleanup codebase and fix any remaining bugs	5		3		
			Fix critical path deadlock issue	8		8		
			[FMS] Improve map performance	8		8		
14	**Demo Day**	**AMOS Demo day including the final project release**						

	Sprint	Sprint Theme	Est. Size	Est. Burndown	Real Size	Real Burndown
				223		**209**
	7	**Improvements & Leftovers from Release**	**17**	**206**	**19**	**190**
	8	**UI: Displaying robot status info**	**26**	**180**	**26**	**164**
	9	**Finish basic set-up (UI, FMS, TBs) & Collision avoidance**	**30**	**150**	**25**	**139**
	10	**Multiple turtlebot management**	**37**	**113**	**29**	**110**
	11	**Tests/Add features to new graph**	**37**	**76**	**37**	**73**
	12	**Scheduling robots for recharging**	**35**	**41**	**41**	**32**
	13	**Finalize project**	**41**	**0**	**32**	**0**
	14	**Demo Day**				



Type	Link / reference
Build / Deploy	https://github.com/amosproj/amos2022ss03-turtlebot-fleet-management/wiki#builddeploy-documentation
User	https://github.com/amosproj/amos2022ss03-turtlebot-fleet-management/wiki#user-documentation
Design	https://github.com/amosproj/amos2022ss03-turtlebot-fleet-management/wiki#design-documentation

\	Context	Name	Version	License	Link
2x	Hardware	iClebo Kobuki Turtlebot	2		http://kobuki.yujinrobot.com
2x		Raspberry PI	3		https://www.raspberrypi.com/products/raspberry-pi-3-model-b/
	Turtlebot	ROS 2	Galactic Geochelone	BSD	https://docs.ros.org/en/galactic/index.html
		kobuki_ros	custom	BSD	https://github.com/kobuki-base/kobuki_ros/issues/41
		mqtt_bridge	custom	MIT	https://github.com/groove-x/mqtt_bridge/issues/55
		Sick LiDAR Localization Software	2	Proprietary	https://www.sick.com/de/en/localization-and-positioning-solutio
		Sick LiDAR Localization Software ROS driver	5.3.0	Apache	https://github.com/SICKAG/sick_lidar_localization
		Sick Map Engineering Tool		Proprietary	
	Fleetmanagement Backend	Flask	2.1.2	BSD	https://github.com/pallets/flask/
		paho-mqtt	1.6.1	EPL, EDL	https://github.com/eclipse/paho.mqtt.python
		matplotlib	3.5.2	PSF	https://github.com/matplotlib/matplotlib
		python-astar	0.93	BSD	https://github.com/jrialland/python-astar
		Shapely	1.8.2	BSD	https://github.com/shapely/shapely
	Fleetmanagement Frontend	Vue.js	3.2.37	MIT	https://github.com/vuejs/core
		Bootstrap	4.6.1	MIT	https://github.com/twbs/bootstrap
		Chartjs	3.8.0	MIT	https://github.com/chartjs/Chart.js
		Chartjs-plugin-annotation	1.4.0	MIT	https://github.com/chartjs/chartjs-plugin-annotation
	Networking between Fleet management and Turtlebot	OpenWRT	22.03	GPLv2	https://openwrt.org/

Last Name	First Name	Value					
Vogler	**Tim**	3		**2.17**	**NOK**		
Scherbel	**Sebastian**	1					
Blöcher	**Meike**	3					
Markert	**Niklas**	2					
Moorthy	**Venkatesh Kumar**	2		0	No size		
Janjua	**Muhammad Usman**	2		1	Trivial size		
				2	Small size		
				3	Medium size		
				5	Large size		
				8	Very large size		
				13	Too large (size)		