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 (white)	...
Team T-shirt (black)	https://www.shirtinator.de/loadBasket/Gip4U1-D_O7
Additional materials	...

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	TBD	Ekaterina Alekseenko
8	2022-06-15			Umang Ramaiya/ Jonas Petersen	Everyone else	TBD	Ekaterina Alekseenko
9	2022-06-22			Umang Ramaiya/ Jonas Petersen	Everyone else	TBD	Ekaterina Alekseenko
10	2022-06-29	Yes		Umang Ramaiya/ Jonas Petersen	Everyone else	TBD	Ekaterina Alekseenko
11	2022-07-06			Umang Ramaiya/ Jonas Petersen	Everyone else	TBD	Ekaterina Alekseenko
12	2022-07-13			Umang Ramaiya/ Jonas Petersen	Everyone else	TBD	Ekaterina Alekseenko
13	2022-07-20	Yes		Umang Ramaiya/ Jonas Petersen	Everyone else	TBD	Ekaterina Alekseenko
14	2022-07-27		Demo day!	Umang Ramaiya/ Jonas Petersen	Everyone else	TBD	Ekaterina Alekseenko
15	2022-08-03		Retrospective	Umang Ramaiya/ Jonas Petersen	Everyone else	TBD	Ekaterina Alekseenko

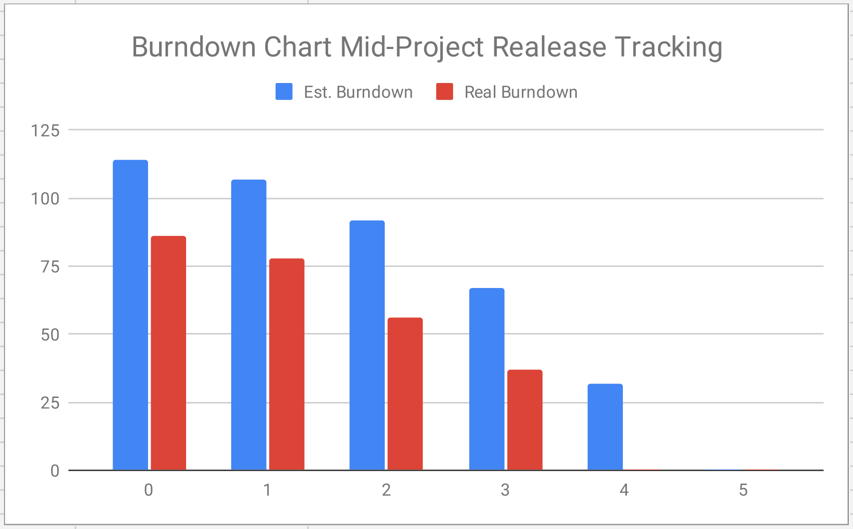
Product Vision	Project Mission
The vision is to have a management system which helps fulfill daily tasks in a smart and intelligent way. TurtleBots are automated guided vehicles (AGVs) which assist humans without their intervention. The TurtleBot fleet management system is envisioned to bring intelligence 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
Sick LiDAR LOC	A software for determining the position of automated guided vehicles (AGVs)
SMET	Sick Map Engineering Tool: helps create maps for localization
ROS	Robot Operating System, a framework that helps researchers and developers build and reuse code between robotics applications
MQTT	A lightweight, publish-subscribe network protocol that transports messages between devices
VDA5050	A standardized interface for AGV communication
RasPi	Raspberry Pi: A credit-card sized computer who's OS acts as a powerful combination to create smart robots
Docker	A software platform that allows you to build, test, and deploy applications quickly
FMS	Fleet management system:
VMap Ingress Module for FMS	Reads scanned LIDAR map file and converts it to FMS Python graph
Graph Module for FMS	Defines a storage format and provides functions for traversal
VDA5050 Module for FMS	To create VDA5050 JSON strings and read VDA5050 packets
MQTT Module for FMS	For communication between FMS and robots
Worker Module for FMS	with info about current location, state and speed it calculates the direction and speed to reach destination
Webserver Module for FMS	To recieve and execute requests from User Interface
Connection Module of Robot	Establishes MQTT connection with broker
Worker Module for Robot	Has a subscriber & publisher, a calculation function and a driver segment
Subscriber (Robot worker module)	Gets relevant topics from the mqtt node on the robot, the localisation topic and the line measurement topic
Publisher (Robot worker module)	Publishes the current location, speed, direction and battery status
Calculation function (Robot worker module)	Calculates the direction and speed for the robot to reach the 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.

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down
0			**Total**		**114**		**86**	**86**
1	**Familiarize with Project**	**Getting to know team and industry partner, project organization**			**7**		**8**	**78**
			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**	**56**
			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**	**37**
			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**	**0**
			[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**			**32**		**0**	0
			Develop the subscriber and publisher (for Worker module)	5				
			Develop the calculation function (for Worker module)	3				
			Develop the driver segment (for Worker module)	5				
			[FMS] Graph route finding algorithm	8				
			[FMS/TB] tests	5				
			Create build process video	3				

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down
			[FMS] VDA5050 packet reciever	3				

Sprint	Sprint Theme	Est. Size	Est. Burndown	Real Size	Real Burndown
0			114		86
1	Familiarize with Project	7	107	8	78
2	Initial setup	15	92	22	56
3	First setup of TurtleBots	25	67	19	37
4	Developing FMS Modules	35	32	37	0
5	Development of TurtleBot worker modules and further developments for FMS modules	32	0	0	0



#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	Theme	Goal	Feature Name	Est. Size (Feature)	Est. Size (Sprint)	Real Size (Feature)	Real Size (Sprint)	Burn-Down

#	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

Type	Link / reference

\	Context	Name	Version	License	Comment
2x	Hardware	iClebo Kobuki Turtlebot	-		http://kobuki.yujinrobot.com
2x	Hardware	RaspberryPI	-		
	Record and generate a map with the use of the turtlebots	SICK Map Engineering Tool	-	Protected	
	Turtlebot	SICK LiDAR Localization Software	-	Protected	https://github.com/SICKAG/sick_lidar_localization
	Turtlebot	ROS 2 (Robot Operating System)	galactic		https://docs.ros.org/en/galactic/index.html
	Turtlebot	turtlebot2_ros2			https://github.com/wn1980/turtlebot2_ros2
	Fleet Management Software	Python	?		
	FleetMgmnt/graph_search.py	python-astar	0.93	BSD 3-Clause	https://github.com/jrialland/python-astar
	UI Terminal	JavaScript	?		
	UI Terminal	Vue.js	?	MIT	
	UI Terminal	Bootstrap	?	MIT	
	Deployment	Docker	?		
	Networking	OpenWRT			https://openwrt.org/

Last Name	First Name	Value					
#REF!	#REF!			5.00	OK		
Vogler	Tim	5					
Scherbel	Sebastian	5					
Blöcher	Meike	5					
Markert	Niklas	5		0	No size		
Janjua	Muhammad Usman	5		1	Trivial size		
Moorthy	Venkatesh Kumar	5		2	Small size		
				3	Medium size		
				5	Large size		
				8	Very large size		
				13	Too large (size)		
		3					